



30日学懂工业机器人

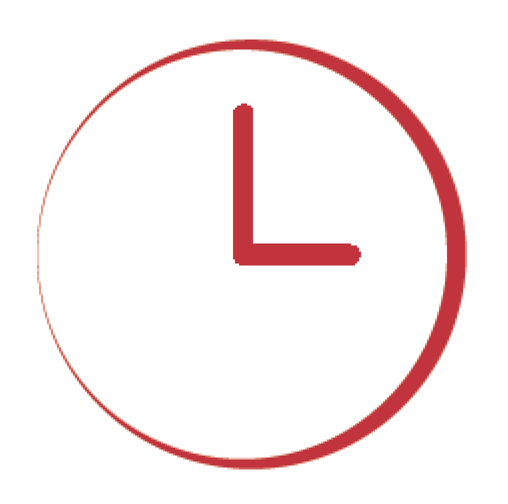
黄风 编著

30天
研究项目案例，
学习视觉追踪等高级应用

20天
全面掌握编程指令和
参数设置

10天
学会编制工作程序

3天
使机器人动起来



清华大学出版社

30 日学懂工业机器人

黄　风　编著

清华大学出版社
北　京

内 容 简 介

本书以三菱工业机器人为基础详细介绍了学习工业机器人应掌握的知识，主要内容如下：工业机器人区别于一般运动控制器所特有的功能；工业机器人编程语言及各指令的详细应用；机器人的工作状态变量及其使用方法；编程所使用的函数及其用法；机器人参数的含义及其设置方法；应用案例十余例。

虽然工业机器人的应用范围越来越广泛，但对初学者来说，还是很神秘。本书按照“由浅入深”“循序渐进”的原则，按 30 个工作日，每日安排适当的学习内容和实验内容，解决了初学者面对茫茫资料无从下手不知所措的尴尬问题。这样经过 3 天的学习，学习者就可以使机器人动起来；经过 10 天的学习，学习者就可以编制一般的工作程序；在学习 20 天之后，学习者可以比较全面地掌握机器人的编程指令、参数设置等知识；经过 30 天的学习，学习者可以研究项目案例，学习视觉追踪等机器人的高级应用。

图书在版编目(CIP)数据

30 日学懂工业机器人/黄风编著.—北京：清华大学出版社，2019
ISBN 978-7-302-47599-6

Ⅰ. ①3…　Ⅱ. ①黄…　Ⅲ. ①工业机器人—基本知识　Ⅳ. ①TP242.2

中国版本图书馆 CIP 数据核字(2017)第 153460 号

责任编辑： 黄　芝　薛　阳
封面设计： 甘陵丰
责任校对： 焦丽丽
责任印制： 刘海龙

出版发行： 清华大学出版社
网　　址： http://www.tup.com.cn，http://www.wqbook.com
地　　址： 北京清华大学学研大厦 A 座　　**邮　　编：** 100084
社 总 机： 010-62770175　　**邮　　购：** 010-62786544
投稿与读者服务： 010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈： 010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载： http://www.tup.com.cn，010-62795954
印 装 者： 三河市君旺印务有限公司
经　　销： 全国新华书店
开　　本： 185mm×260mm　　**印　　张：** 26.5　　**字　　数：** 642 千字
版　　次： 2019 年 7 月第 1 版　　**印　　次：** 2019 年 7 月第1 次印刷
印　　数： 1～2000
定　　价： 79.50 元

产品编号：073565-01

前　　言

近年来，工业机器人在制造业领域的应用如火如荼，工业机器人是智能制造的核心技术。本书从实用的角度出发，对工业机器人的选型、系统集成、实用配线、特殊功能、编程语言、状态变量、参数功能及设置、工业机器人与PLC触摸屏的综合应用、工业机器人与视觉系统的综合应用等方面做了深入浅出的介绍，提供了大量的程序指令解说案例。

本书根据实际的应用成果，介绍了工业机器人在抛光、测量分拣、码垛、视觉追踪等方面的应用。这些应用是实际工作的总结，可以给机器人的设计、集成、编程、调试人员以实用的参考，也可以使高校学生在学习一定的基础知识后，了解如何在实际的项目中配置和设计机器人集成项目，建造从学校到实际应用的一条快速走廊。

本书以新颖的编排形式，遵循"由浅入深""由少到多"的原则，按30个工作日，分步对工业机器人的知识进行实用性的介绍，解决了初学者面对茫茫资料不知所措的问题。本书以一种品牌的机器人为主进行系统的学习。经过系统的学习之后，读者就可以上手做工程项目，所以本书不是泛泛而谈的科普性书籍。

本书第1～19章是工业机器人的基础理论介绍，主要介绍机器人的选型、配线、编程指令、状态变量、函数、参数功能及设置。本书从最实用的角度来介绍如何选型，各技术指标的含义，以及如何配线。在第3章就指导读者让机器人动起来，消除工业机器人的神秘感。

本书对机器人编程指令进行了详细讲解。经过对编程指令由浅入深的学习，读者可以牢固地掌握这些指令。在第16～18章"参数功能及设置"中，结合软件的使用对重点参数的功能及设置做了说明。这也是从使用者的角度着想的。

从第20章开始，介绍工业机器人在实际工程项目中的应用，且每章介绍一个工程项目。重点介绍面对客户的要求，如何提出解决方案，如何配置系统硬件，如何宏观地分析工作流程和绘制工作流程图，如何编制机器人的相关程序。本书提供的应用案例将对工程技术人员有很大的帮助。

本书第26～29章介绍了机器人视觉追踪的理论、指令应用、参数设置和编程实例，这是机器人应用比较高端的部分。

申建北、胡彬、辛曼一、吴建发、杨桂珍、胡伟、刘勇、黄纯、付芩等参加了本书的编写。

感谢林步东对本书提供的大量支持。

由于作者学识有限，书中不免有疏漏和不足之处，希望读者提出批评建议，与作者进行交流。

黄　风

2018年11月

目　　录

第 1 章　第 1 日——工业机器人的构成和技术指标

【学习目的】

在第 1 日的学习中，要学习关于工业机器人的最基本的知识，学习工业机器人的技术规格以及相关的性能指标。这是因为在使用工业机器人时最初遇到的就是选型问题，即客户提出要求，我们应该如何为客户的项目选型？选型中最重要的技术指标是什么？这些技术指标的含义是什么？

1.1　什么是工业机器人

工业机器人实质上是一套由运动控制器控制，可以实现多轴联动的多关节型工业机械。

我们通常看到的数控机床也是由运动控制器控制的工业机械，但是机器人与数控机床的区别是机器人是多关节型工业机械（模仿人类动作），而且机器人一般可以做到 6 轴联动，而数控机床大部分是 3 轴联动。

1.2　工业机器人的构成

工业机器人由以下几个主要部分构成，如图 1-1 所示。

1. 机器人本体

本体包含机械构件（各关节工作臂以及减速机）和伺服电机。伺服电机已经安装在机械本体上。

2. 控制器

控制器包括控制 CPU、伺服驱动器、基本 I/O、各种通信接口（USB/以太网）。

3. 示教单元

示教单元也称为“手持操作器”，简称 TB，用于手动操作机器人运行，确定各工作点、JOG 运行、设置参数、设置原点、显示机器人工作状态。

4. 附件

附件包括抓手和各种接口板。

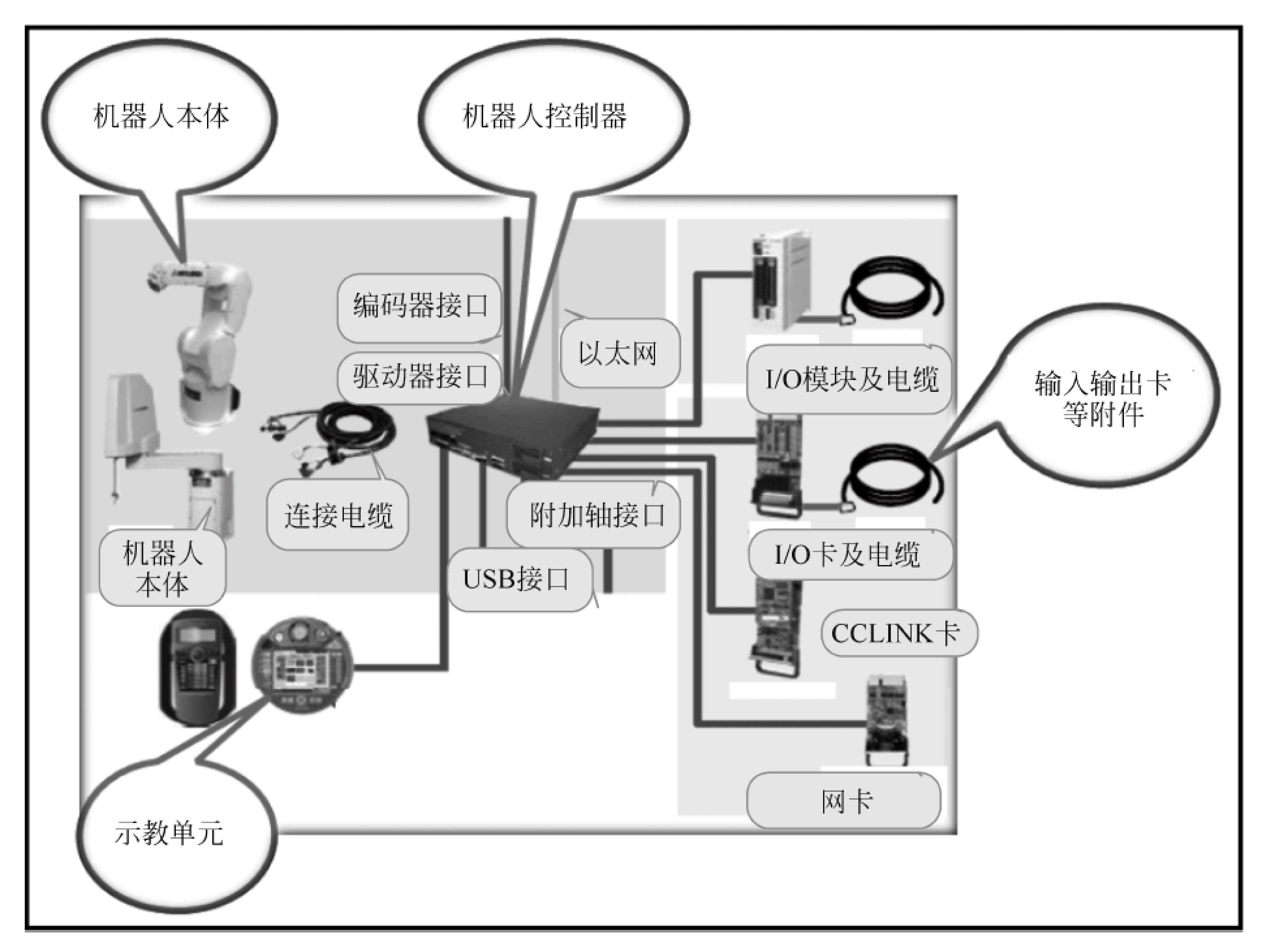


图 1-1　工业机器人系统构成简图

1.3　工业机器人技术规格

1.3.1　垂直多功能机器人技术规格

如表 1-1 所示为垂直多功能机器人技术规格。在该技术规格中，标明了伺服电机容量、动作范围、最大合成速度、搬运重量，这些都是机器人选型的重要依据。

表 1-1　垂直多功能机器人技术规格

型号		单位	规格			
			RV-4F	RV-4FL	RV-7F	RV-7FL
环境规格			未标注：一般；C:清洁；M:油雾			
动作自由度			6	6	6	6
安装方式			落地、吊顶、挂壁			
结构			垂直多关节			
驱动方式			AC 伺服电机/带全部轴制动			
位置检测方式			绝对值编码器			
电机容量	J1	W	400		750	
	J2		400		750	
	J3		100		400	
	J4		100		100	
	J5		100		100	
	J6		50		50	

续表

型号		单位	规格			
动作范围	J1	deg	480			
	J2		240		−115～125	−110～130
	J3		0～161	0～164	0～156	0～162
	J4		±200	±200	±200	±200
	J5		±120			
	J6		±360			
最大速度	J1	deg/s	450	420	360	288
	J2		450	336	401	321
	J3		300	250	450	360
	J4		540		337	
	J5		623		450	
	J6		720			
最大动作半径		mm	514.5	648.7	713.4	907.7
最大合成速度		mm/s	9000		11 000	
搬运重量		kg	4	4	7	7
位置重复精度		mm	±0.02			
循环时间		s	0.36		0.32	0.35
环境温度		℃	0～40			
本体重量		kg	39	41	65	67
允许力矩	J4	N·m	6.66		16.2	
	J5		6.66		16.2	
	J6		3.90		6.86	
允许惯量	J4	$kg \cdot m^2$	0.20		0.45	
	J5		0.20		0.45	
	J6		0.10			

1.3.2 水平多功能机器人技术规格

如表 1-2 所示为水平多功能机器人技术规格。在该技术规格中，标明了臂长、动作范围、最大合成速度、搬运重量、位置重复精度，这些都是机器人选型的重要依据。水平多功能机器人多用于平面搬运和垂直搬运。

表 1-2 水平多功能机器人技术规格

型号	单位	规格		
		RH-6FH35 ** /M/C	RH-6FH45 ** /M/C	RH-6FH55 ** /M/C
环境规格		未标注：一般；C:清洁；M:油雾		
动作自由度		4	4	4
安装方式		落地		
结构		水平多关节		
驱动方式		AC 伺服电机		
位置检测方式		绝对值编码器		

续表

型号		单位	规格		
臂长	NO1 臂长	mm	125	225	325
	NO2 臂长		225		
			100		400
			100		100
			100		100
			50		50
动作范围	J1	deg	340		
	J2		290		
	J3	mm	＊＊＝20：200　＊＊＝34：340		
	J4	deg	720		
最大速度	J1	deg/s	400		
	J2		670		
	J3	mm/s	2400		
	J4	deg/s	2500		
最大动作半径		mm	350	450	550
最大合成速度		mm/s	6900	7600	8300
搬运重量		kg	最大6(额定3)		
位置重复精度		mm	±0.010		
循环时间		s	0.29		
环境温度		℃	0～40		
本体重量		kg	36	36	37
允许惯量	额定	$kg\cdot m^2$	0.01		
	最大		0.12		

1.4　对工业机器人主要技术指标的解释

1.4.1　机器人部分技术规格名词术语

(1) 安装位置：机器人的可安装方式，如落地、吊顶、挂壁。

(2) 驱动方式：机器人各轴的动力源。一般采用AC伺服电机。

(3) 位置检测：检测机器人各轴运行位置的器件。采用绝对位置编码器。

(4) 动作范围：J1～J6轴以“度数”为单位。

(5) 最大速度：J1～J6轴以“度/秒”为单位。

(6) 最大动作半径：在基本坐标系内，控制点的动作半径范围。以mm为单位(以机械IF坐标原点为控制点)。

(7) 最大合成速度：指控制点在X-Y-Z方向上的最大矢量速度。

1.4.2　机器人的“动作自由度”

要确定一个刚体(一个三维物体，而不是一个点)在空间中的位置，首先需要在该刚体上

选择一个点并指定该点的位置，因此需要 3 个坐标数据来确定该点的位置。但是，即使位置点已确定，仍有无数可能用来确定物体相对于所选“位置点”的“形位”（因为刚体还可以绕“位置点”做三维旋转）。为了完全定位空间物体的“形位”，除了确定物体上“位置点”的位置外，还须确定该物体的“形位”。根据空间几何学分析，需要 6 个数据才能完全确定刚体物体的位置和“形位”。基于同样的理由，就需要有 6 个自由度才能将物体放置到空间中的期望位置上。因此为抓取和传送在空间不同位置和“形位”的物件，传送机构也应具有 6 个自由度。每一个自由度就是在一个维度上运动的能力。机械手的自由度越多，表示其在空间定位的能力越强。

机械手的每一个自由度是由其独立驱动关节来实现的，所以在实际应用中，关节和自由度在表达机械手的运动灵活性方面是意义相同的。又由于关节在实际结构上是由回转轴或移动轴组成的，所以习惯称为“轴”。因此，就有 6 自由度、6 关节或 6 轴机械手的命名方法。它们都说明某机械手的操作有 6 个独立驱动的关节结构，能在工作空间实现达到任意位置和“形位”。如果是 4 轴机器人就表示有 4 个自由度。6 轴机器人的动作及自由度如图 1-2 所示。

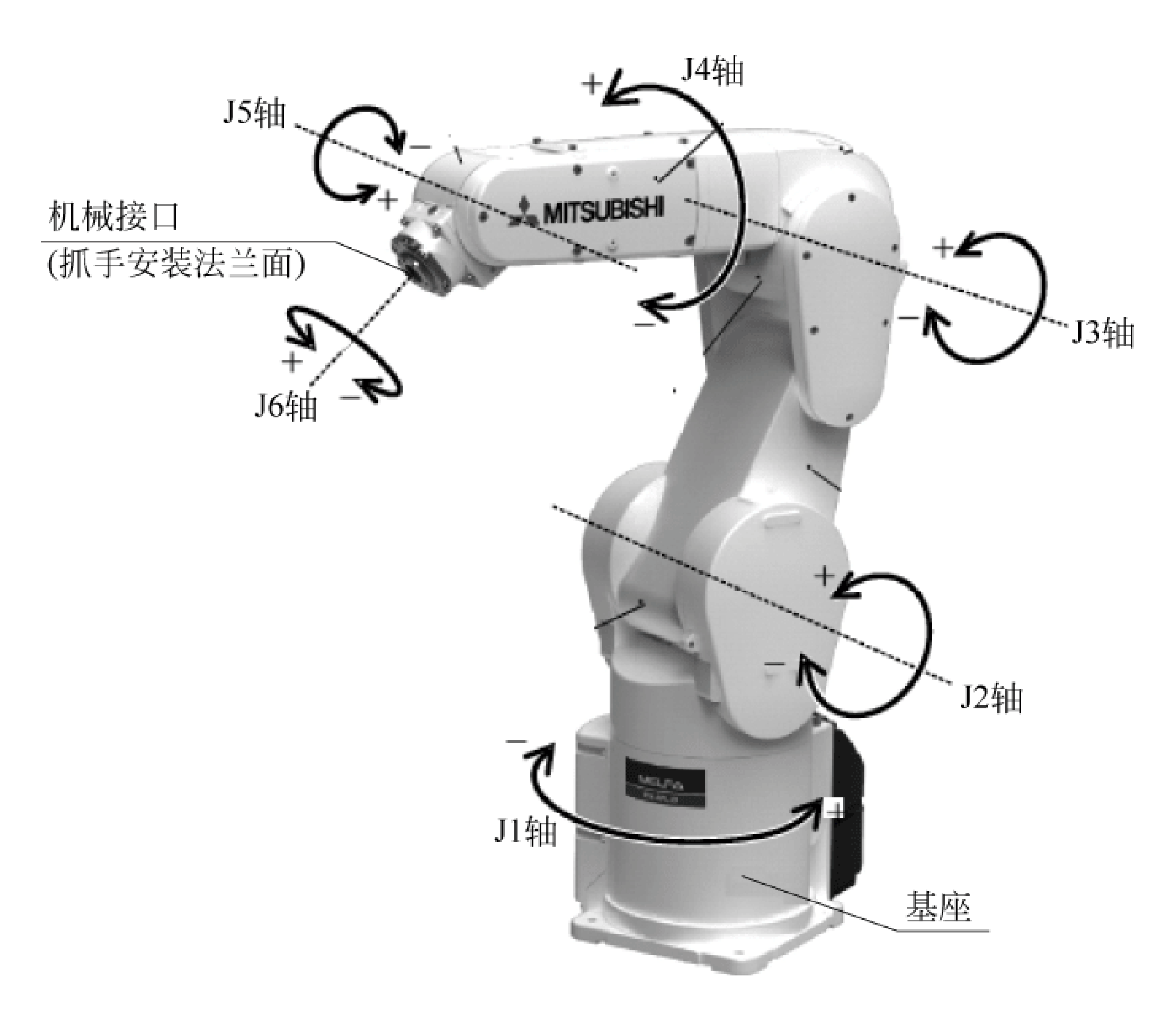


图 1-2　6 轴机器人的动作及自由度

1.4.3　机器人动作的“最大速度”

机器人动作的最大速度有以下两种表示方法。

（1）用每一轴的最大角速度表示，如表 1-3 所示。

（2）用机器人的控制点（即最前端法兰的中心点）移动的最大线速度表示，如表 1-3 所示。

这两个指标在机器人技术规格中已经规定（各型号指标不同）。

表 1-3　机器人的最大角速度和最大线速度

轴	速度类型	RV-13F	RV-20F
J1 轴	deg/s	290	110
J2 轴		234	110
J3 轴		312	110
J4 轴		375	124
J5 轴		375	125
J6 轴		720	360
最大合成速度	mm/s	10 450	4200

在自动程序中设置速度时，通常以最大速度为基准，设置速度倍率——百分数，获得实际速度。

1.4.4　机器人的“最大动作半径”

在基本坐标系内，控制点的最大动作半径范围就是“最大动作半径”，以 mm 为单位（以机械 IF 坐标原点为控制点），如图 1-3 和图 1-4 所示。

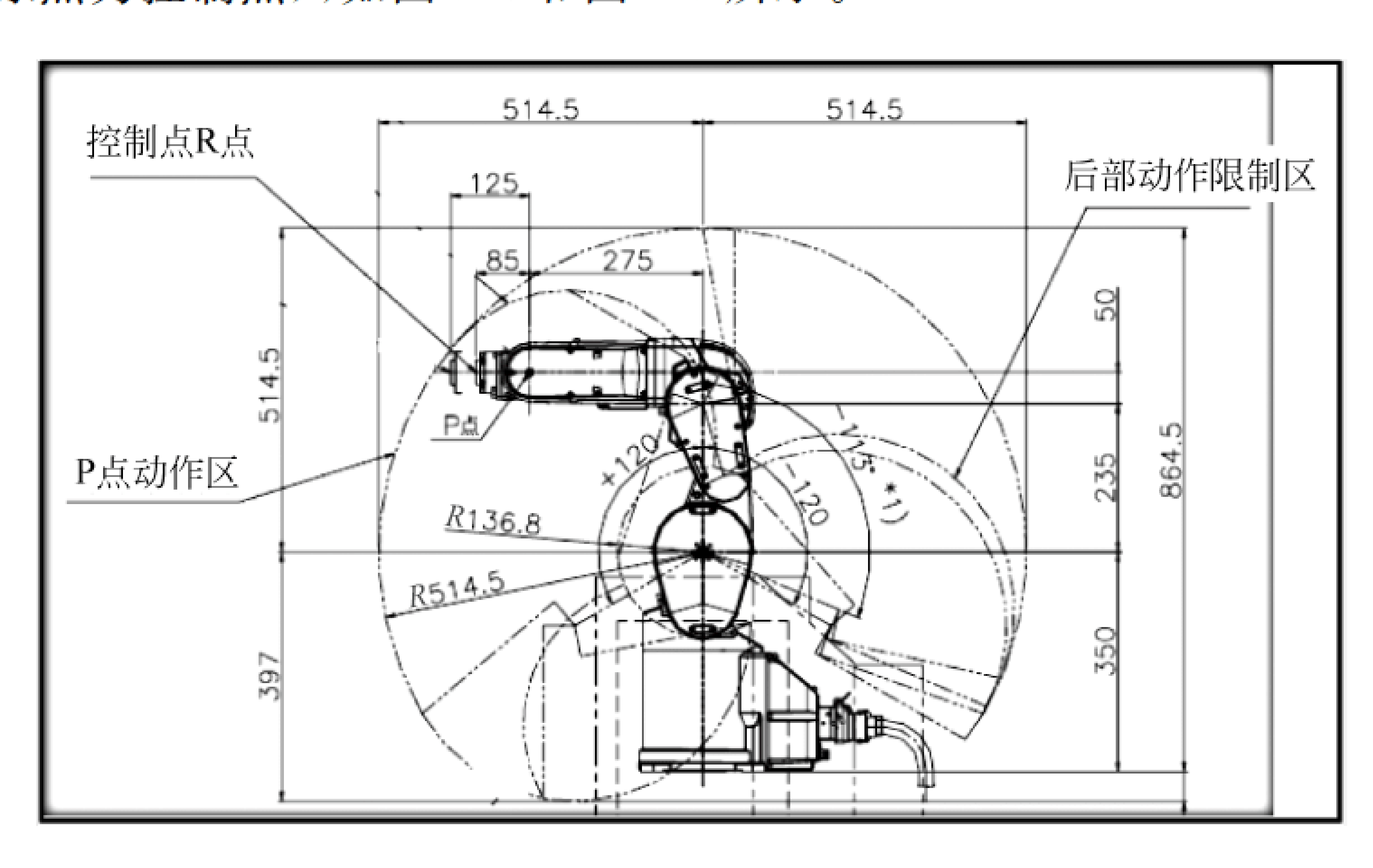


图 1-3　控制点的最大动作半径范围主视图

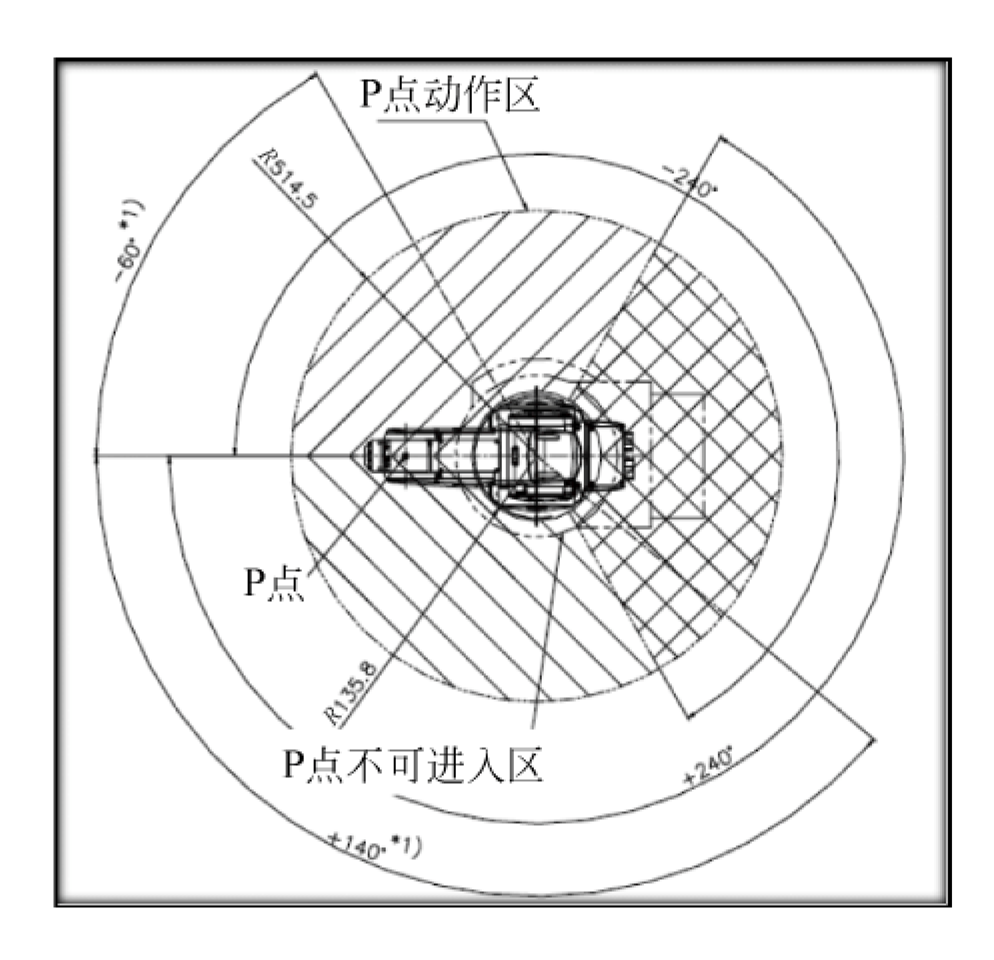


图 1-4　控制点的最大动作半径范围俯视图

1.4.5 机器人的“可搬运重量”

“可搬运重量”是指机器人以额定速度运行不发生报警的能搬运移动物体的重量，以 kg 为单位。“可搬运重量”在技术规格上有规定，是选型时的重要指标。部分型号的机器人可搬运重量如表 1-4 所示。

表 1-4 部分型号的机器人可搬运重量

型号	RV-2F	RV-4F	RV-7F	RV-13F	RV-20F
可搬运重量/kg	2	4	7	13	20

如图 1-5 所示为机器人在搬运重物。



图 1-5 机器人搬运重物

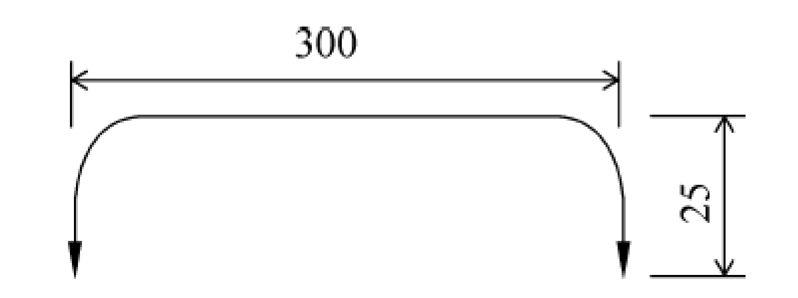


图 1-6 测试机器人重复定位精度的轨迹

1.4.6 机器人的“位置重复精度”

机器人的“位置重复精度”指机器人夹持额定重量工件，以高速动作模式(MvTune2)，按如图 1-6 所示的轨迹反复运行时的定位精度。在对定位精度有要求的场合，这个指标就显得很重要。

一般机器人重复定位的精度可达到 0.02mm。

1.5 控制器技术规格

如表 1-5 所示为控制器技术规格一览表。控制器技术规格有控制轴数、存储容量、可控制的输入输出点数、可使用电源范围、内置接口等。

表 1-5 控制器技术规格一览表

项目		单位	规　格	备　注
型号			CR751-Q，CR751-D	
控制轴数			最多 6 轴	
存储容量	示教位置数	点	39 000	
	步数	步	78 000	
	程序个数	个	512	
编程语言			MELFA-BASIC V	
位置示教方式			示教方式或 MDI 方式	

续表

项目		单位	规　　格	备　　注
外部输入输出	输入输出	点	输入点/输出点	最多可扩展至256/256
	专用输入输出	点	分配到通用输入输出中	STOP1点为固定
	抓手开闭输入输出	点	输入8点/输出8点	内置
	紧急停止输入	点	1	冗余
	门开关输入	点	1	冗余
	可用设备输入	点	1	冗余
	紧急停止输出	点	1	冗余
	模式输出	点	1	冗余
	机器人出错输出	点	1	冗余
	附加轴同步	点	1	冗余
	模式切换开关输入	点	1	冗余
接口	RS-422	端口	1	TB专用
	以太网	端口	1	
	USB	端口	1	
	附加轴接口	通道	1	SSCNET3 与MR-J3-B,MR-J4-B连接
	跟踪接口	通道	2	连接编码器
	选购件插槽	插槽	2	连接选购件I/O
电源	输入电源范围	V	RV-4F系列： 单相AC(180～253)V RV-7F/13F系列： 三相AC(180～253)V或 单相AC(207～253)V	
	电源容量	kVA	RV-4F系列：1.0 RV-7F系列：2.0 RV-13F系列：3.0	
	频率	Hz	50/60	

控制器技术规格名词术语如下。

1. 第1部分

存储容量：指用示教单元可以示教确认的位置点数量。

步数：指一个程序内的“步数”，例如78 000步。

程序个数：指同时可以存放在控制器内的程序数量。

编程语言：MELFA-BASIC V。

位置示教方式：是用示教单元驱动机器人本体，对当前位置进行记录的方式。

MDI方式：MDI是Manual Data Input的缩写，是将数值直接输入以确定“位置点”的方式。

外部输入输出：指通过使用外部I/O单元或模块可扩展的输入输出点数量，例如I265/O256。

专用输入输出：指由控制器内部已经定义的输入输出的功能。

抓手开闭输入输出：专门用于控制抓手的输入输出点，例如I8/O8。

RS-422 通信口：控制器内置的串行通信口。TB(示教单元)专用。

2. 第 2 部分

以太网端口：控制器内置的以太网通信口。

USB 接口：控制器内置的 USB 通信口，用于计算机与机器人连接。

附加轴接口：控制器内置的通信口，用于 SCNET Ⅲ 与 MR-J3-B、MR-J4-B 系列伺服驱动器的连接。

采样接口：控制器内置编码器信号接口，用于接收来自外部编码器的信号，在视觉追踪等场合经常使用。

选购件插槽：控制器内置的插口，用于安装外部 I/O 卡。

输入电压范围：控制器使用的电压范围。

(1) RV-4F 系列：AC(180～253)V。

(2) RV-7F/13F 系列：三相 AC(180～253)V 或单相 AC(207～253)V。

电源容量 kVA：

(1) RV-4F 系列：1.0。

(2) RV-7F 系列：2.0。

(3) RV-13F 系列：3.0。

1.6 需要思考的问题

(1) 什么是机器人的合成速度？

(2) 什么是机器人的搬运重量？

(3) 如何测定机器人的重复定位精度？

第2章 第2日——工业机器人的安装和连接

【学习目的】

在第2日的学习中，要学习机器人的实际安装和连接，了解机器人和控制器各接口的名称和功能，了解如何连接“急停信号”和“模式选择信号”，如何连接“启动信号”和“停止信号”，如何初步定义外部I/O卡各输入输出端子的功能。只有将机器人系统连接起来，才能使机器人运动。

2.1 工业机器人各部分的名称及用途

三菱垂直型6轴机器人各部分名称如图2-1所示。

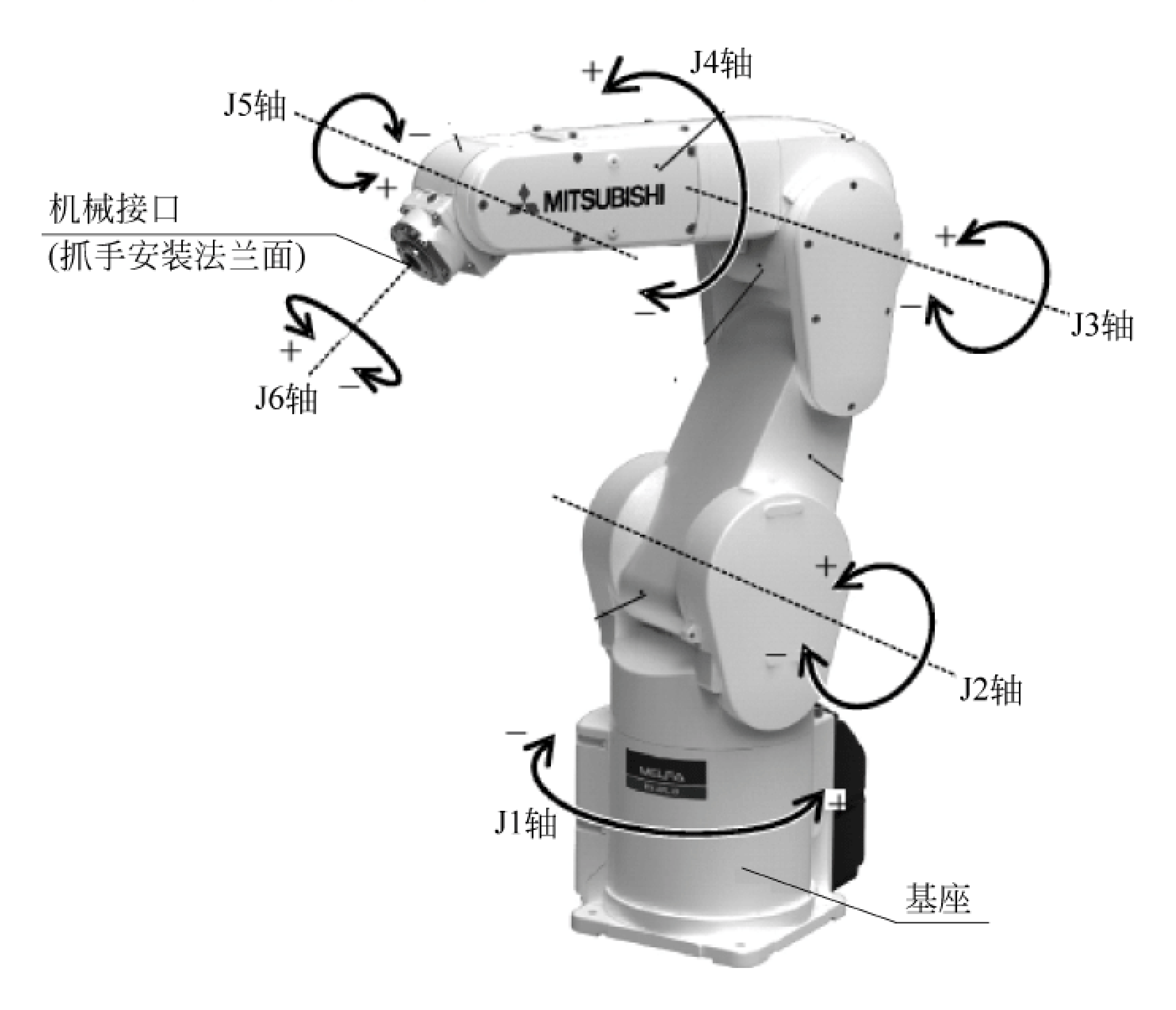


图2-1 机器人各部分名称及用途

本节以三菱垂直型6轴机器人为例，说明各部分的名称及用途。

(1) 基座：基座是安装机器人的机械构件。基座的中心点就是机器人基本坐标系的原点。垂直型机器人可以有落地式、吊顶式、壁挂式等多种安装方式。

(2) 各轴旋转方向：J1轴、J2轴、J3轴、J4轴、J5轴、J6轴各自在空间的旋转方向如图2-1所示。

(3) 抓手安装法兰面：抓手安装法兰面在 J6 轴上，用于安装抓手。法兰面的中心就是 TOOL 坐标系的原点。

2.2 控制器各接口的说明

机器人控制器的接口如图 2-2 所示。

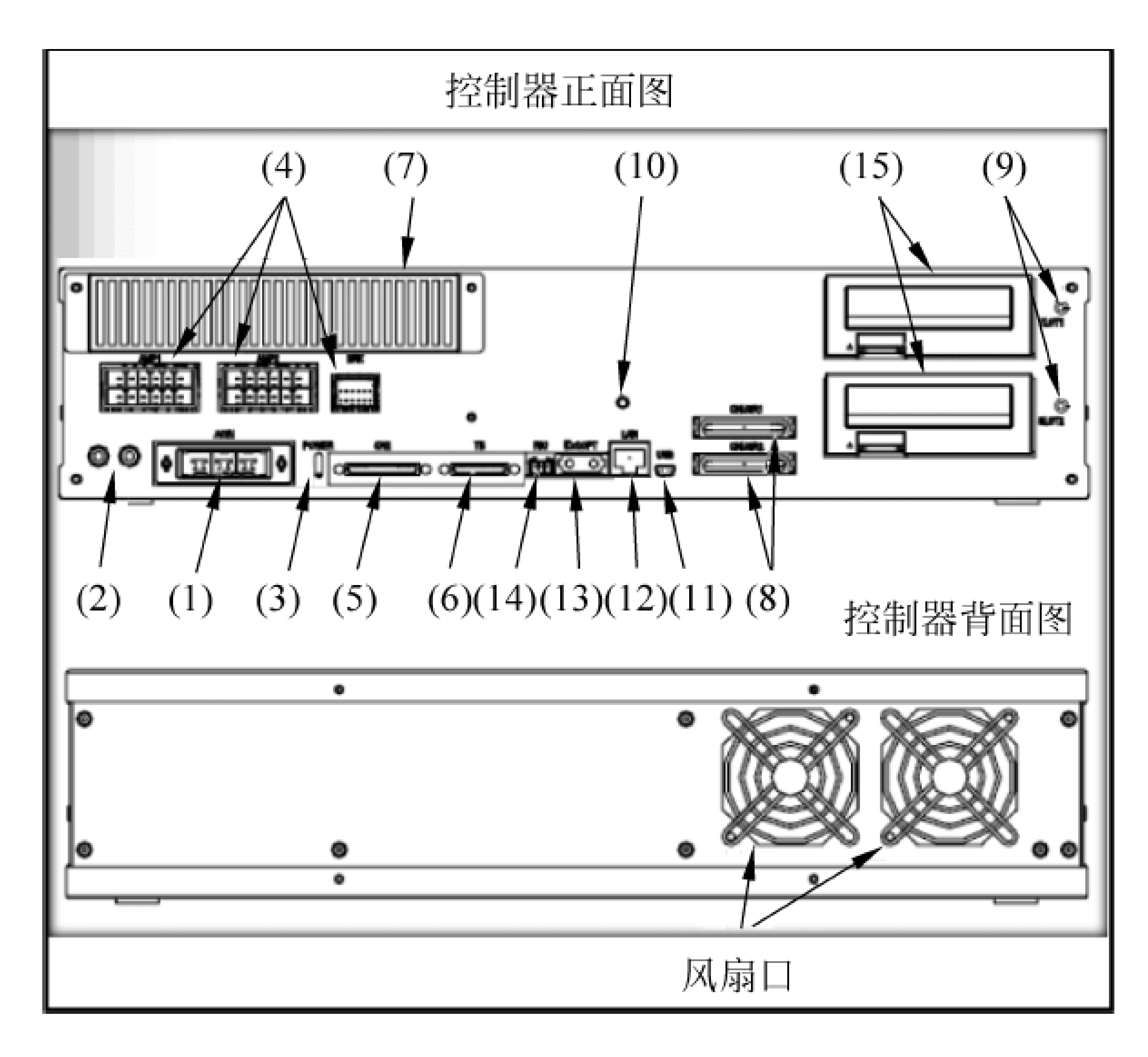


图 2-2 控制器接口示意图

1. 控制器各部分接口名称及用途

在机器人系统中，机器人本体与控制器是分离的，就像在数控机床中机床本体与控制器分离一样。本节以 CR751-D(独立型)控制器为例，说明控制器各接口的作用。

对 CR751-D 控制器接口及其功能的说明如下。

(1) ACIN 连接器——AC 电源(单相，AC200V)输入用插口。

(2) PE 端子——接地端子。

(3) POWER 指示灯——控制电源 ON/OFF 指示灯。

(4) 电机电源连接插口 AMP1、AMP2——电机电源用插口。BRK：电机制动器插口。

(5) 电机编码器连接插口 CN2——电机编码器插口。

(6) 示教单元连接插口(TB)——R33TB 连接专用(未连接示教单元时需要安装标配插头)。

(7) 过滤器盖板——空气过滤器、电池安装两用。

(8) CNUSR 插口——机器人专用输入输出插口(附带插头)。

(9) 接地端子。

(10) 充电指示灯——确认拆卸盖板时的安全(防止触电)指示灯。当机器人伺服＝ON 使得控制器内的电源基板上积累电能时，本指示灯亮灯(红色)。关闭控制电源经过一定时间(几分钟)后灯熄灭。

(11) USB 插口——USB 连接用。

(12) LAN 插口——以太网连接插口。

(13) Export 插口——附加轴连接用插口。

(14) RIO 插口——扩展输入输出模块用插口。

(15) 选购件插槽——选购件卡安装用插槽。

2.3 机器人与控制器连接

机器人本体与控制器的连接如图 2-3 所示。

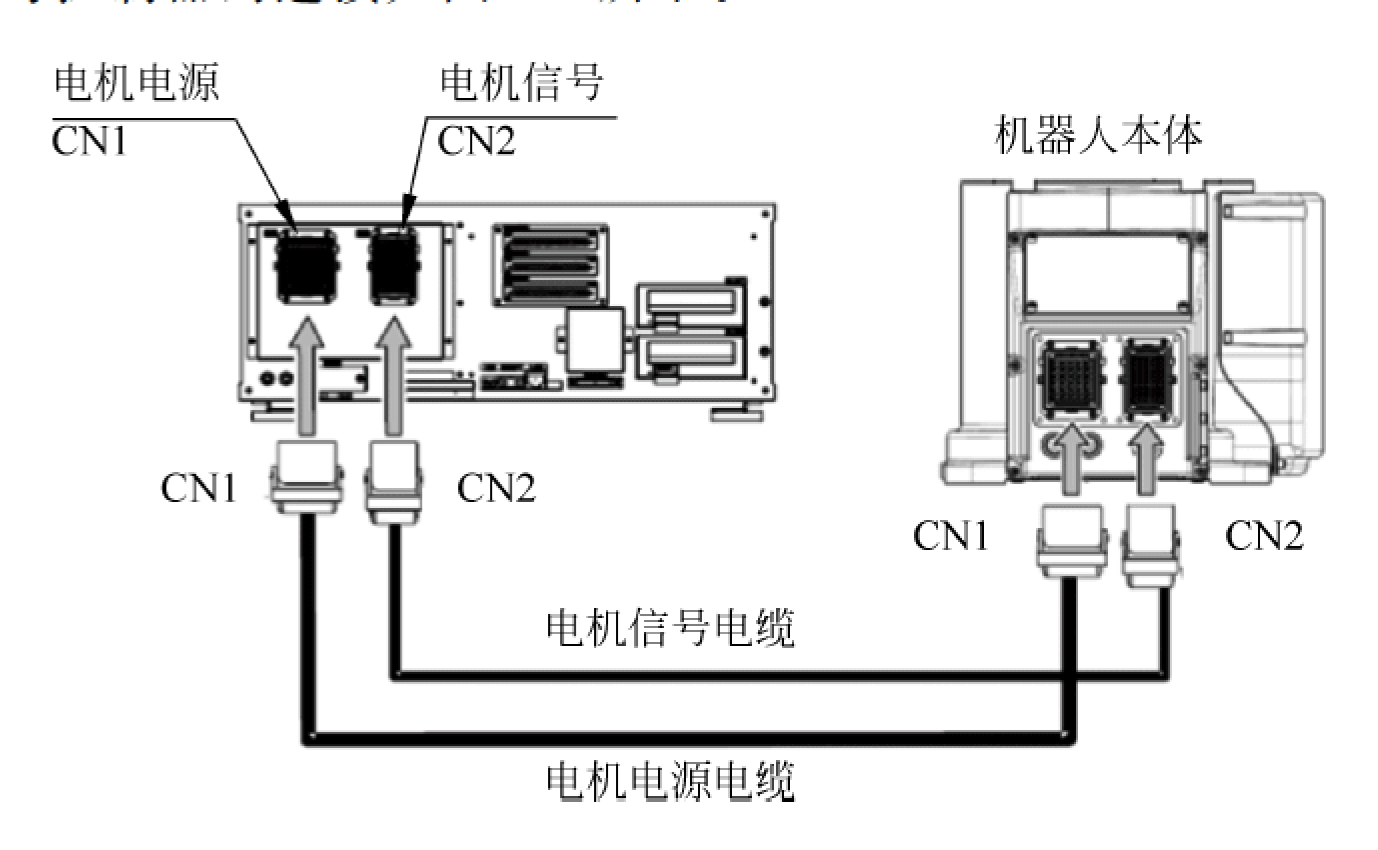


图 2-3 机器人本体与控制器的连接

机器人本体与控制器主要通过两条电缆连接。

(1) 电源电缆——通过 CN1 连接。

(2) 编码器反馈电缆——通过 CN2 口连接。

2.4 机器人与外围设备连接

1. 控制器电源连接

电源电缆属于标配。根据机器人型号不同,使用单相 220V 电源或者使用三相 220V 电源。需要使用一个能够提供三相 220V 的变压器。变压器的容量应该是“控制器规格一览表”中电源容量要求的 1.2～1.5 倍。

不能直接使用工厂里的三相 380V 电源,否则会立即烧毁控制器。

在主电源回路应该接入“断路器”和“空开”。

2. 控制器与 GOT 的连接

通过以太网口连接。

3. 控制器与计算机的连接

可以通过以太网口连接,也可以通过 USB 口连接,实际使用中多通过 USB 连接。

2.5 急停及安全信号

“外部急停开关”和“门保护开关”的接线如图 2-4 所示。

这些开关信号都接入 CNUSR1 接口。CNUSR1 接口是控制器标配接口,如图 2-5 所示。

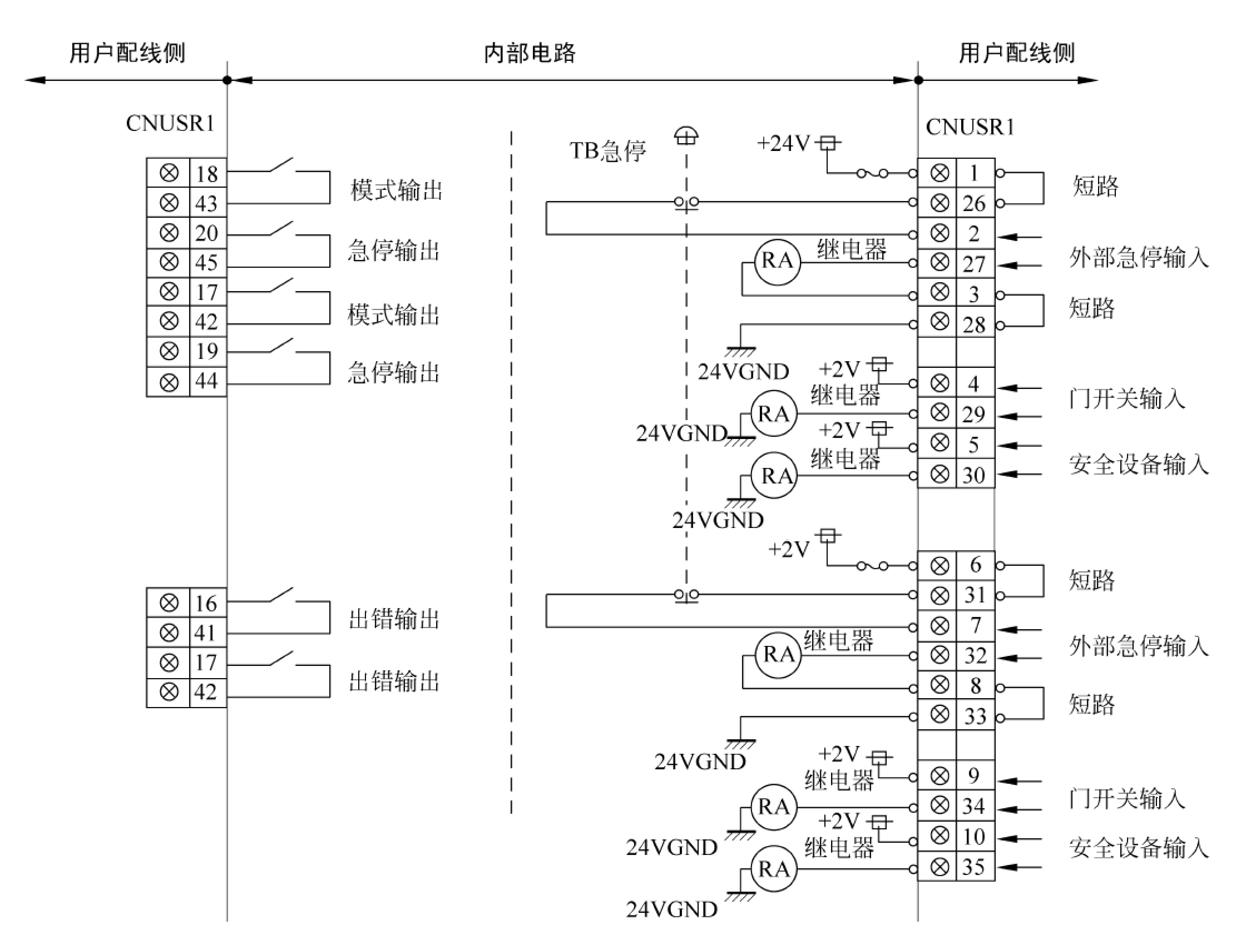


图 2-4　外部安全开关的配线

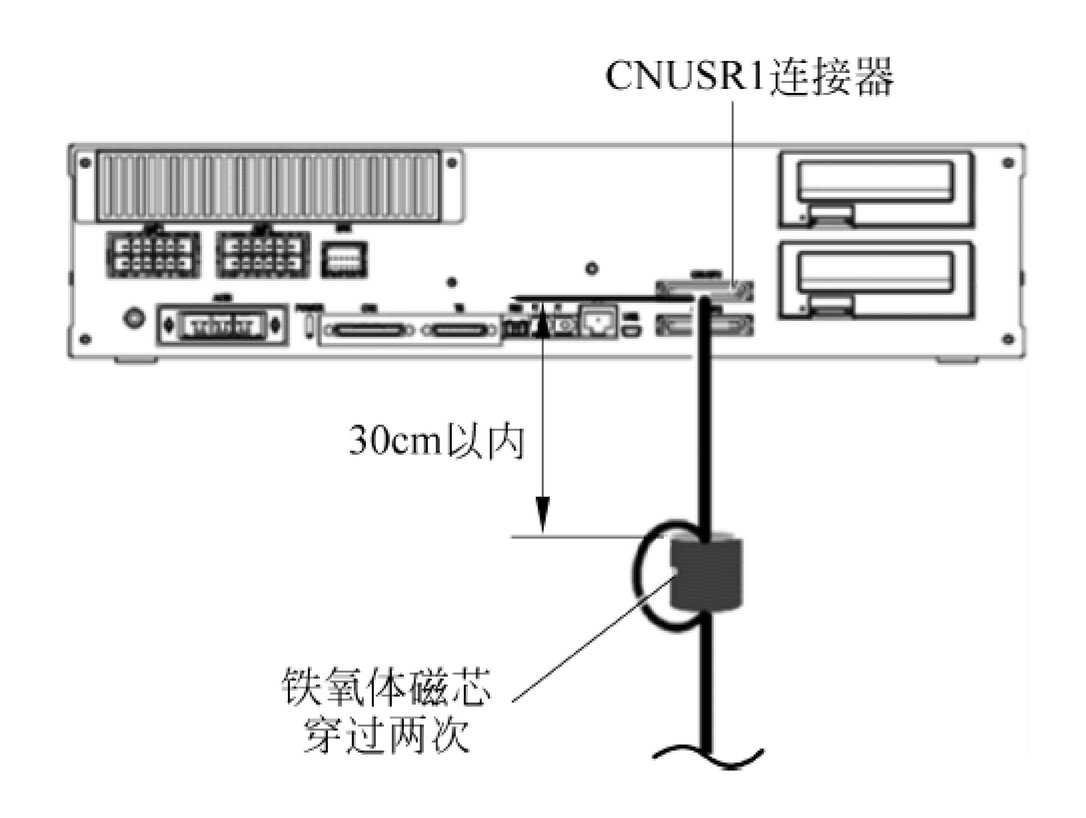


图 2-5　从控制器的 CNUSR1 接口引出的“特别输入输出信号”

1. 外部急停开关

外部急停开关一般指安装在操作面板上的急停开关。当然，急停开关可以装在生产线的任何必要部位。外部急停开关采用 B 接点冗余配置，如图 2-4 所示。外部急停开关在 CNUSR1 插口引出电缆的 2～37 和 7～32 端子之间。

所谓冗余配置是指在配线时必须使用双触点型“急停开关”，保证即使在一个触点失效时，另外一个触点也能够切断“急停回路”。

2. 门开关

门开关用于检测工作门的开启关闭状态。门开关采用B接点冗余配置。在正常状态下，门保护开关的功能是在设备的防护门被打开时使机器人伺服系统=OFF，停止动作起到安全保护作用。设备的门打开以后，机器人停止运行，以免出现伤人事故。门开关的功能是使伺服=ON/OFF。

"门开关"在CNUSR1插口引出电缆的4～29和9～34端子之间。

所谓冗余配置是指在配线时必须使用双触点型"开关"，保证即使在一个触点失效时，另外一个触点也能够切断"门开关回路"。

门保护开关必须为"常闭型"。门打开时，门保护开关=OFF。

自动运行时：门打开→伺服停止→报警。

解除：关门→复位→伺服ON→启动。

3. 安全辅助(可用设备)开关

安全辅助开关功能：对示教作业进行保护。如果在示教作业中出现异常，按下"安全辅助开关"，能够使伺服=OFF，停止机器人运动。安全辅助开关采用B接点冗余配置。"安全辅助开关"在CNUSR1插口引出电缆的5～30和10～35端子之间，也是冗余配置。

4. 跳跃信号

SKIP信号是跳跃信号，当SKIP=ON时，则立即停止执行当前程序行，跳到"指定的程序行"。SKIP信号端子在CNUSR2口的9～34。SKIP信号的接法如图2-6所示。

项　　目		规　　格	内部电路
型式		DC输入	
输入点数		1	
绝缘方式		光电耦合器绝缘	
额定输入电压		DC24V	
额定输入电流		约11mA	
使用电压范围		DC(21.6～26.4)V(波动率5%以内)	
ON电压/ON电流		DC8V以上/2mA以上	
OFF电压/OFF电流		DC4V以下/1mA以下	
输入电阻		约2.2kΩ	
响应时间	OFF→ON	1ms以下	
	ON→OFF	1ms以下	
公共端方式		一点一个公共端	
外线连接方式		连接器	

图2-6　SKIP信号的接法

2.6　模式选择信号

机器人的工作模式有"自动模式"和"手动模式"。"工作模式选择"是指选择机器人的工作模式。

1. 自动模式

通过(操作面板上的)外部信号控制程序"启动"或"停止"。要将"操作权"信号切换为

“外部信号”有效状态。

2. 手动模式

通过“示教单元”的 JOG 模式操作机器人动作。

“工作模式选择”的信号标配在 CNUSR1 接口的规定信号端子 49～24 和 50～25，如图 2-7 及表 2-1 所示（源型接法，24V 电源由控制器提供）。

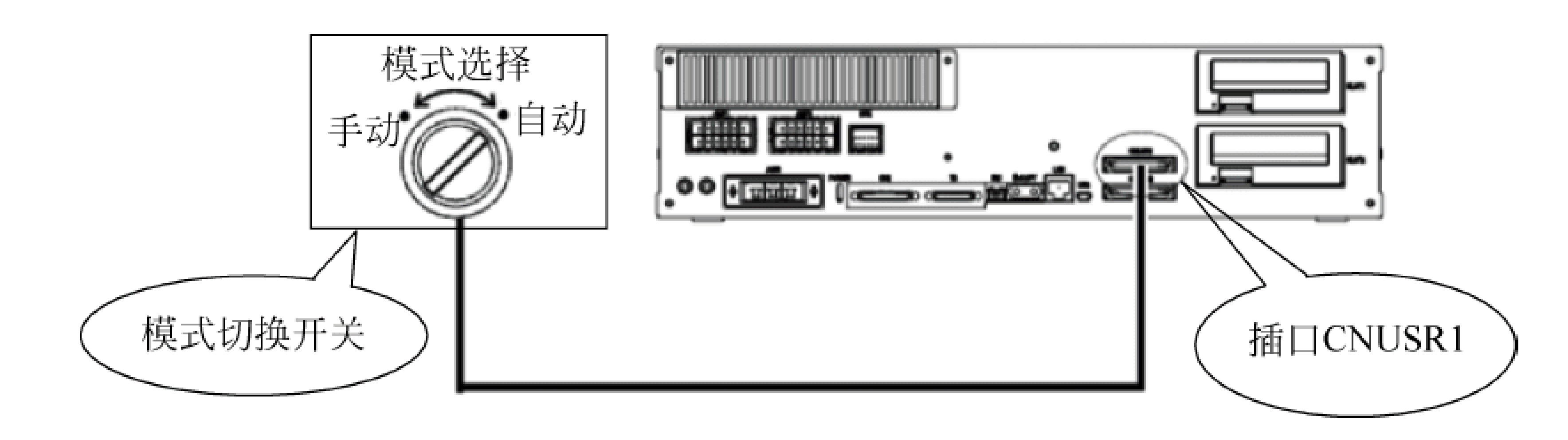


图 2-7　模式选择开关的电缆插口

表 2-1　模式选择开关的针脚编号

插口：CNUSR1		切 换 模 式	
针脚编号	功能	手动模式	自动模式
49	1 输入接点	OFF	ON
24	1 输入电源＋24V		
50	2 输入	OFF	ON
25	2 输入电源＋24V		

2.7　输入输出信号的连接

2.7.1　概述

除了控制器标配的（CNUSR1/CNUSR2）输入输出信号（有急停信号、安全信号、模式选择信号）之外，为了实现更多的控制功能，包括对外部设备的控制和信号检测，实用的机器人系统需要使用更多的 I/O 信号。机器人系统可以扩展的外部 I/O 信号为 256/256 点。扩展外部 I/O 信号的方法可以通过配置“I/O 模块”和“I/O 接口板”来实现。

2.7.2　实用板卡配置

机器人系统配置的“外部 I/O 模块”有“板卡型”和“模块型”两种。

1. 板卡型

（1）板卡型 2D-TZ368、2D-TZ378 可直接插接在控制器的 SLOT1、SLOT2 的插口（32 点输入，32 点输出）。

（2）板卡必须有对应的“站号”。这与一般控制系统相同，只有设置站号，才能分配确定的 I/O 地址。使用“板卡型 I/O”时，站号根据插入的 SLOT 确定。

SLOT1＝站号 1　　SLOT2＝站号 2

2. 模块型

模块型输入输出单元配置有外壳，相对独立，通过专用电缆与控制器连接。

2.7.3 板卡型2D-TZ368（漏型）的输入输出电路技术规格

1. 输入电路技术规格

输入信号的漏型接法如图2-8所示。

（1）输入电压：DC（12～24）V。

（2）输入点数：32点。

（3）公共端方式：32点共一个公共端。

项　目	规　格
形式	DC输入
输入点数	32
绝缘方式	光电绝缘
额定输入电压/额定输入电流	DC 12V/3mA　DC 24V/9mA
使用电压范围	DV(10.2~26.4)V
ON 电压/ON电流	DC 8V以上/2mA以上
OFF电压/OFF电流	DC 4V以下/1mA以下
输入电阻	2.7kΩ
响应时间OFF—ON	10ms 以下
响应时间ON—OFF	10ms 以下
公共端方式	32 点共一个公共端
外线连接方式	连接器

图2-8　输入信号的漏型接法

所谓“公共端COM”是指板卡本身这些输入点的“公共端”。一个板卡上有32个输入点，这些输入点的接法一样，所以就只有一个公共接点（漏型共DC24V+）（在一个回路中，输入模块的“点”视作“负载”）。

（4）漏型/源型接法。

① 开关点与电源正极相连即为源型接法。

② 开关点与电源负极相连即为漏型接法。

2. 输出电路技术规格

输出信号的漏形接法如图2-9所示。

（1）输出形式：晶体管输出。

DC24V电源由外部提供，DC（12～24）V。

项　目	规　格
形式	晶体管输出
输出点数	32
绝缘方式	光电绝缘
额定负载电压	DC 12V/ DC 24V
使用电压范围	DV(10.2~30)V
最大负载电流	0.1A/点
OFF时泄露电流	0.1mA以下
ON最大电压降	DC 0.9V以下
输入电阻	2.7kΩ
响应时间OFF—ON	10ms 以下
响应时间ON—OFF	10ms 以下
额定保险丝	1.6A
公共端方式	16点共一个公共端
外线连接方式	连接器
外部供电电源	DC12/24V 60mA
漏型 +24V/+12V 输出 0V 保险丝	

图 2-9　输出信号的漏型接法

（2）输出点数：32 点。

（3）公共端方式：16 点共一个公共端。

3. I/O 卡 2D-TZ368 与 PLC 输入输出模块的连接

如图 2-10 所示是 2D-TZ368 与 PLC 输入输出模块的连接图。其中，QX41 是 PLC 输入模块，QY81P 是 PLC 输出模块。

2D-TZ368 与 PLC 输入输出模块的连接为漏型接法。

（1）漏型输出电路。在图 2-10 中，由外部 DC24V 电源给输出部分的三极管提供工作电源，所以必须在规定的点接入外部 DC24V 电源。在“电源—开关—负载”回路中，其电流流向是“DC24V＋—负载（QX41）—集电极（TZ368）—发射极（DC0V）”。

（2）漏型输入电路。其流向是“DC24V＋—负载（TZ368）—集电极（QY81P）—发射极（DC0V）”。

在一个标准回路中，输出模块的每一点相当于一个开关。一个板卡上有 32 个输出点，这些输出点的接法一样，所以也有一个公共接点 COM（漏型，共 DC0V）。

如果三极管的发射极接 DC0V，则三极管的“集电极”接“负载”，这就是所谓的“集电极”开路，其公共端就是 DC0V。

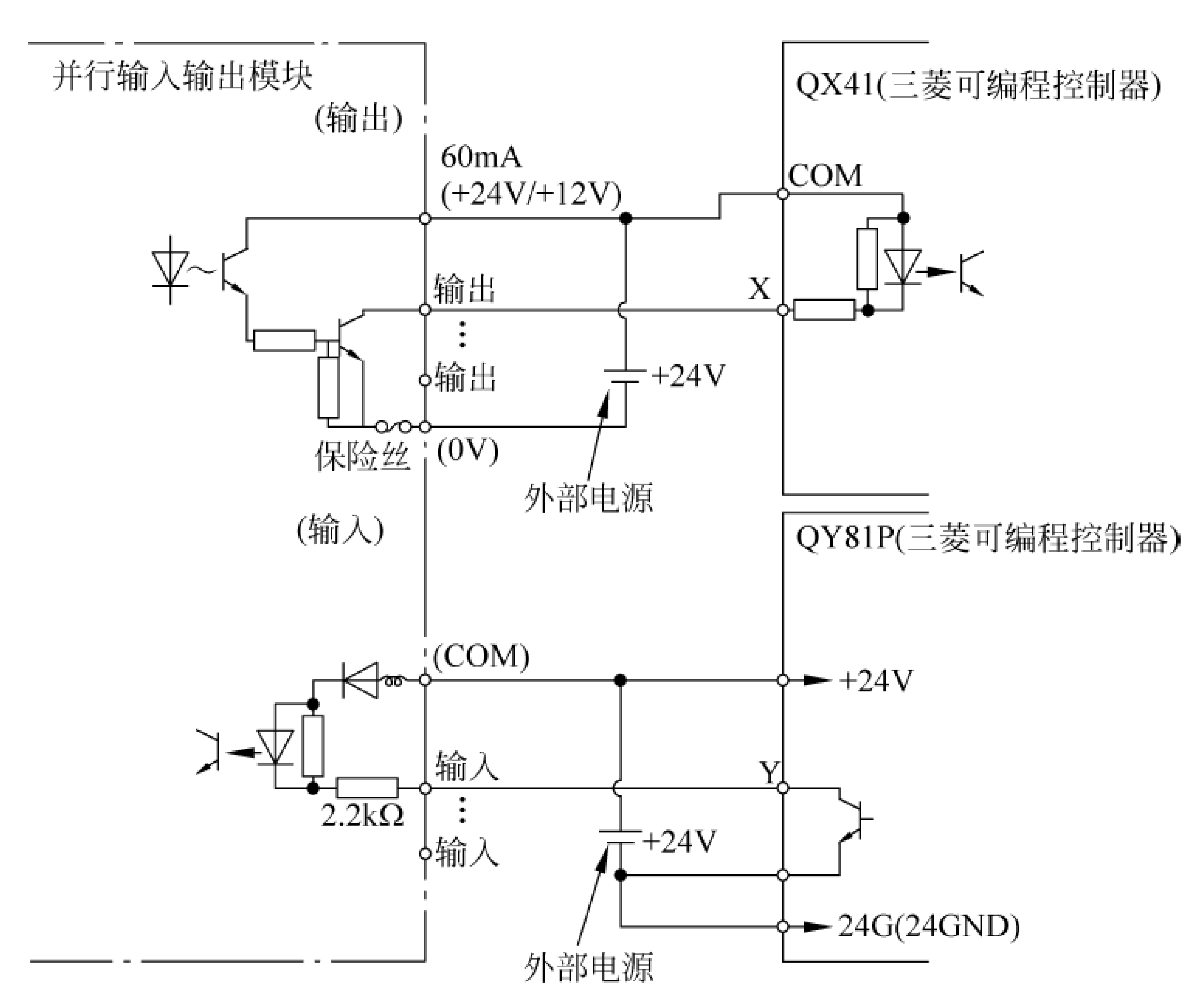


图 2-10　2D-TZ368 与 PLC 输入输出模块的连接图

2.7.4　板卡型 2D-TZ378(源型)的输入输出电路技术规格

1. 输入电路技术规格

输入电路技术规格如图 2-11 所示。

项　目	规　格
形式	DC输入
输入点数	32
绝缘方式	光电绝缘
额定输入电压/额定输入电流	DC 12V/3mA　DC 24V/9mA
使用电压范围	DV(10.2~26.4)V
ON电压/ON 电流	DC 8V以上/2mA以上
OFF电压/OFF电流	DC 4V以下/1mA以下
输入电阻	2.7kΩ
响应时间 OFF—ON	10ms 以下
响应时间 ON—OFF	10ms 以下
公共端方式	32点共一个公共端
外线连接方式	连接器
2.7kΩ　输入端　820Ω　0V　COM端	

图 2-11　输入电路技术规格

(1) 输入电压：DC(12～24)V。

(2) 输入点数：32 点。

(3) 公共端方式：32 点共一个公共端。

2. 输出电路技术规格

输出电路技术规格如图 2-12 所示。

(1) 输出形式：晶体管输出。DC24V 电源由外部提供，DC(12～24)V。

(2) 输出点数：32 点。

(3) 公共端方式：16 点共一个公共端。

项　目	规　格
形式	晶体管输出
输入点数	32
绝缘方式	光电耦合器绝缘
额定负载电压	DC 12V/ DC 24V
使用电压范围	DV(10.2~30)V
最大负载电流	0.1A/点
OFF时泄露电流	0.1mA 以下
ON 最大电压降	DC 0.9V以下
输入电阻	2.7kΩ
响应时间OFF—ON	10ms 以下
响应时间ON—OFF	10ms 以下
额定保险丝	1.6A
公共端方式	16 点共一个公共端
外线连接方式	连接器
外部供电电源	DC12V~24V 60mA
源型	保险丝 +24V/+12V 输出 0V

图 2-12　输出电路技术规格

3. I/O 卡 2D-TZ378 与 PLC 输入输出模块的连接

如图 2-13 所示是 2D-TZ378 与 PLC 输入输出模块的连接图。其中，QX81 是 PLC 输入模块，QY81P 是 PLC 输出模块。

2D-TZ378 与 PLC 输入输出模块的连接为源型接法。

(1) 源型输出电路。在图 2-13 中，由外部 DC24V 电源给输出部分的三极管提供工作电源，所以必须在规定的点接入外部 DC24V 电源。在“电源—开关—负载”回路中，电流的流向是“DC24V+—开关点(TZ378)—负载(QX81)—(DC0V)。”

(2) 源型输入电路。其流向是“DC24V+—开关点(QY81P)—负载(TZ378)—COM(DC0V)”。

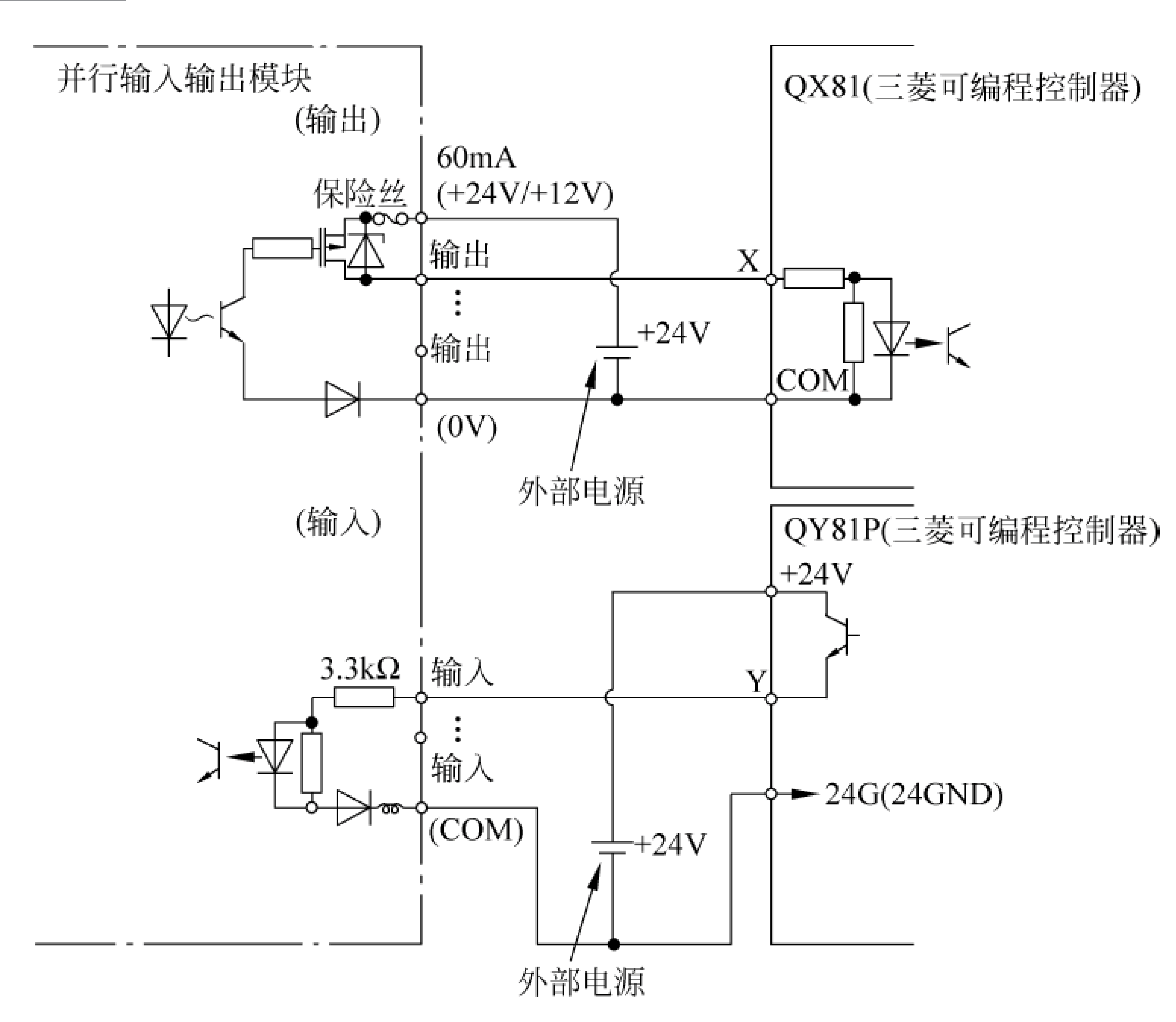


图 2-13　2D-TZ378 与 PLC 输入输出模块的连接图

在实际布线中必须严格分清漏型、源型接法,如果接错会烧毁 I/O 板。

2.7.5　硬件的插口与针脚定义

硬件的插口与针脚定义如下。

I/O 卡 2DTZ-368 安装在控制器的 SLOT1 或 SLOT2 插口中,由连接电缆引出。其针脚分布如图 2-14 和表 2-2 所示。现场连接时,注意电缆颜色与针脚的关系,如表 2-3 和表 2-4 所示。

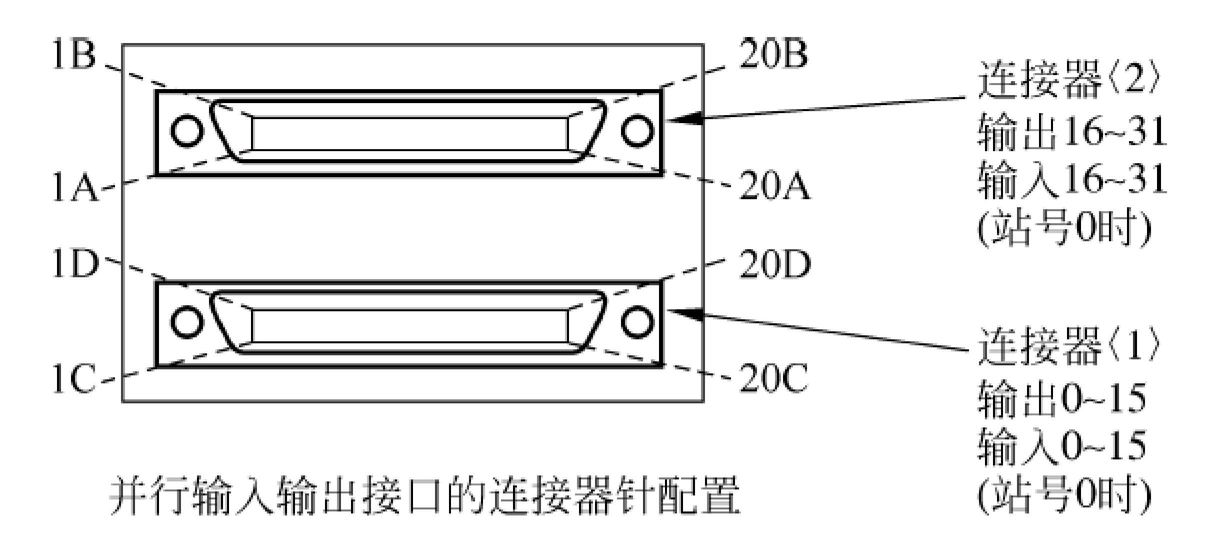


图 2-14　输入输出卡的硬件插口

表 2-2　在各硬件插口内输入输出信号的范围

插槽编号	站　号	通用输入输出编号范围	
		连接器(1)	连接器(2)
SLOT1	0	输入：0～15 输出：0～15	输入：16～31 输出：16～31
SLOT2	1	输入：32～47 输出：32～47	输入：48～63 输出：48～63

表 2-3 插口 1 输入输出信号的针脚与电线颜色的关系

针脚编号	线色	信号名	针脚编号	线色	信号名
1C	橙红 a	0V	1D	橙黑 a	
2C	灰红 a	COM	2D	灰黑 a	
3C	白红 a	空端子	3D	白黑 a	
4C	黄红 a	空端子	4D	黄黑 a	
5C	桃红 a	通用输入 15	5D	桃黑 a	通用输出 15
6C	橙红 b	通用输入 14	6D	橙黑 b	通用输出 14
7C	灰红 b	通用输入 13	7D	灰黑 b	通用输出 13
8C	白红 b	通用输入 12	8D	白黑 b	通用输出 12
9C	黄红 b	通用输入 11	9D	黄黑 b	通用输出 11
10C	桃红 b	通用输入 10	10D	桃黑 b	通用输出 10
11C	橙红 c	通用输入 9	11D	橙黑 c	通用输出 9
12C	灰红 c	通用输入 8	12D	灰黑 c	通用输出 8
13C	白红 c	通用输入 7	13D	白黑 c	通用输出 7
14C	黄红 c	通用输入 6	14D	黄黑 c	通用输出 6
15C	桃红 c	通用输入 5	15D	桃黑 c	通用输出 5
16C	橙红 d	通用输入 4	16D	橙黑 d	通用输出 4
17C	灰红 d	通用输入 3	17D	灰黑 d	通用输出 3
18C	白红 d	通用输入 2	18D	白黑 d	通用输出 2
19C	黄红 d	通用输入 1	19D	黄黑 d	通用输出 1
20C	桃红 d	通用输入 0	20D	桃黑 d	通用输出 0

表 2-4 插口 2 输入输出信号的针脚与电线颜色的关系

针脚编号	线色	信号名	针脚编号	线色	信号名
1A	橙红 a	0V	1B	橙黑 a	
2A	灰红 a	COM	2B	灰黑 a	
3A	白红 a	空端子	3B	白黑 a	
4A	黄红 a	空端子	4B	黄黑 a	
5A	桃红 a	通用输入 31	5B	桃黑 a	通用输出 31
6A	橙红 b	通用输入 30	6B	橙黑 b	通用输出 30
7A	灰红 b	通用输入 29	7B	灰黑 b	通用输出 29
8A	白红 b	通用输入 28	8B	白黑 b	通用输出 28
9A	黄红 b	通用输入 27	9B	黄黑 b	通用输出 27
10A	桃红 b	通用输入 26	10B	桃黑 b	通用输出 26
11A	橙红 c	通用输入 25	11B	橙黑 c	通用输出 25
12A	灰红 c	通用输入 24	12B	灰黑 c	通用输出 24
13A	白红 c	通用输入 23	13B	白黑 c	通用输出 23
14A	黄红 c	通用输入 22	14B	黄黑 c	通用输出 22
15A	桃红 c	通用输入 21	15B	桃黑 c	通用输出 21
16A	橙红 d	通用输入 20	16B	橙黑 d	通用输出 20

续表

针脚编号	线　色	信　号　名	针脚编号	线　色	信　号　名
17A	灰红 d	通用输入 19	17B	灰黑 d	通用输出 19
18A	白红 d	通用输入 18	18B	白黑 d	通用输出 18
19A	黄红 d	通用输入 17	19B	黄黑 d	通用输出 17
20A	桃红 d	通用输入 16	20B	桃黑 d	通用输出 16

各电缆的接法如图 2-15 所示。

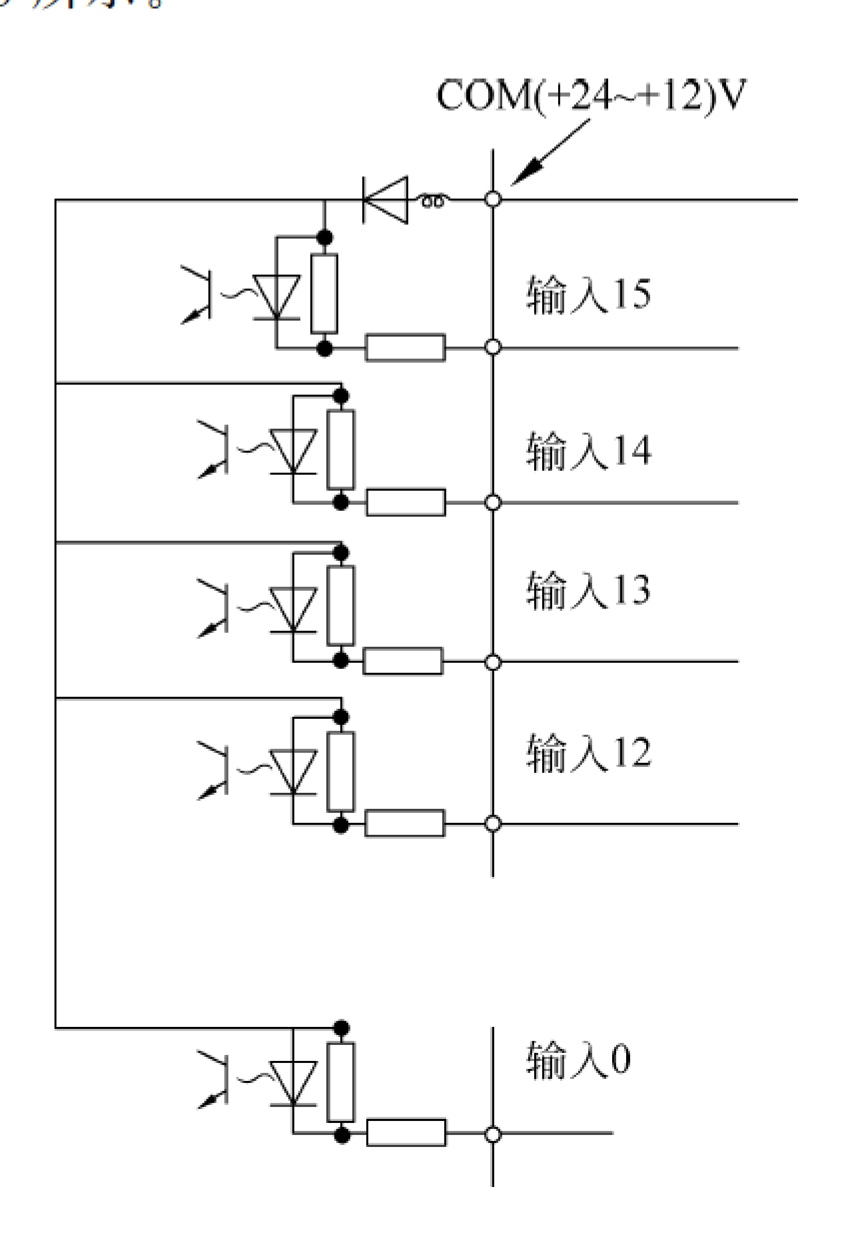


图 2-15　各输入端子的连接方法

2.7.6　输入输出模块 2A-RZ361

输入输出模块 2A-RZ361，有外壳，类似于较为独立的模块。每一模块和板卡都必须设置“站号”。这与一般控制系统相同，只有设置站号，才能分配确定的 I/O 地址。

2.8　实用机器人控制系统的构建

一套实用的机器人控制系统的构建如图 2-16 所示。

1. 主回路电源系统

1）电源等级

在主回路系统中必须特别注意：机器人使用的电源为单相 220V 或三相 220V，不是工厂现场使用的三相 380V。要根据机器人的型号确定其电源等级。

使用三相 220V 电源时，需要专门配置三相 220V 变压器。

2）主要安全保护元件

在主回路中应该配置：无熔丝断路器，接触器。

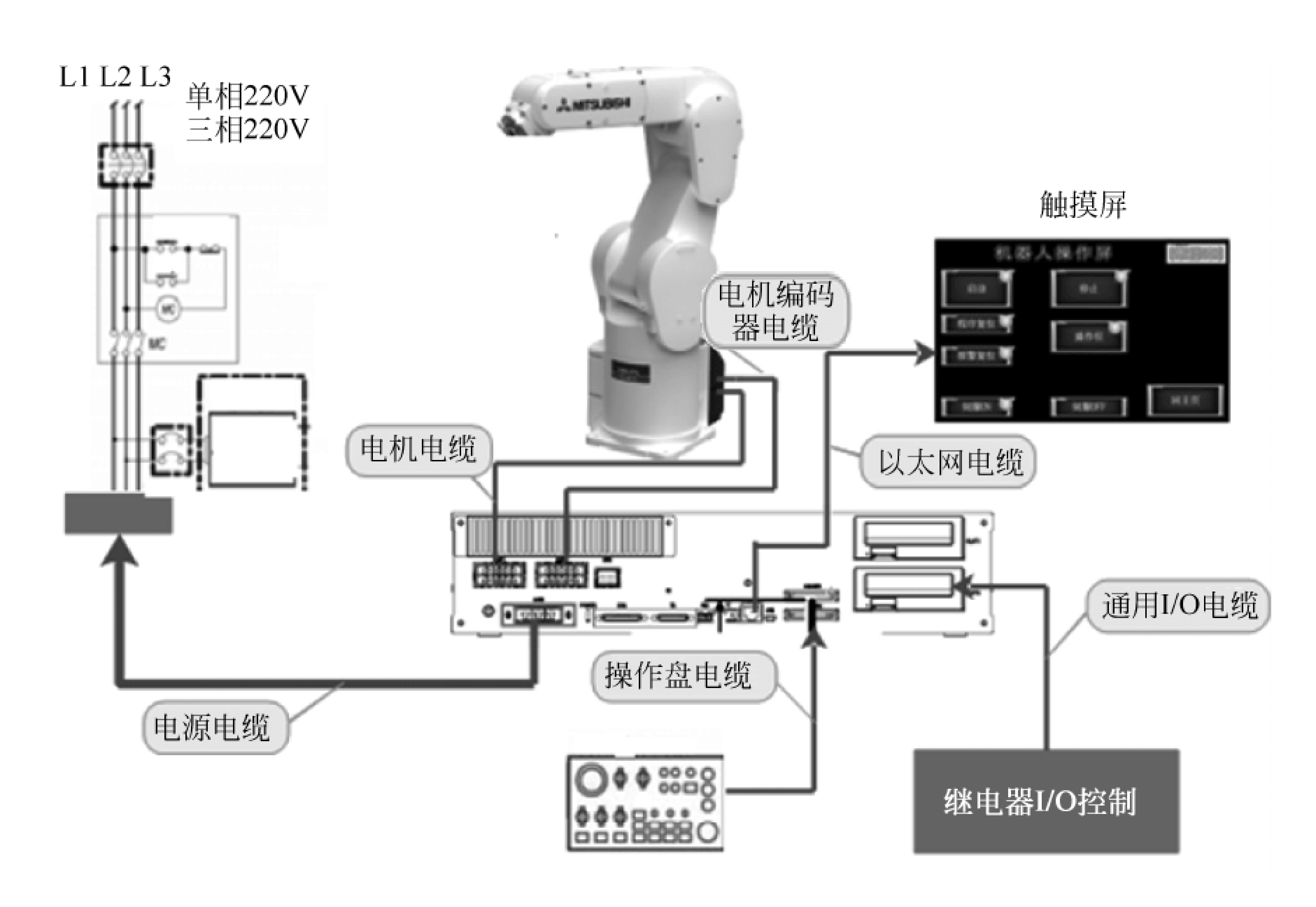


图 2-16　实用机器人控制系统的构建

3）专用电缆

在机器人控制器一侧，有专用的电源插口。出厂时配置有电源电缆，如果长度不够，用户可以将电缆加长。

4）控制电源

在主回路中再接入一个“控制变压器”。控制变压器提供 DC24V 直流电源，可以供操作面板和外围 I/O 电路使用。

2. 控制器与机器人本体连接

伺服电机的电源电缆和伺服电机编码器的电缆是机器人的标配电缆。注意，CN1 口是电机电源电缆插口，CN2 口是电机编码器电缆插口。

3. 操作面板与控制器的连接

操作面板由用户自制，至少包括以下按钮。

(1) 电源 ON；

(2) 电源 OFF；

(3) 急停；

(4) 工作模式选择(选择型开关)；

(5) 伺服 ON；

(6) 伺服 OFF；

(7) 操作权；

(8) 自动启动；

(9) 自动停止；

(10) 程序复位；

(11) 程序号设置(波段选择开关)；

(12) 程序号确认。

这些信号来自于控制器的不同插口,如表 2-5 所示。

表 2-5 工作信号及其插口

序 号	按钮名称	对应插口
1	电源 ON	主回路控制电路
2	电源 OFF	主回路控制电路
3	急停	控制器 CNUSR1 插口
4	工作模式选择	控制器 CNUSR1 插口
5	伺服 ON	SLOT1 中 I/O 板 2D-TZ368
6	伺服 OFF	
7	操作权	
8	自动启动	
9	自动停止	
10	程序复位	
11	程序号选择	
12	程序号确认	

在配线时要分清强电和弱电(电源等级),分清是源型接法还是漏型接法,如果接法错误会烧毁设备。

4. 外围检测开关和输出信号

SLOT1 中 I/O 板 2D-TZ368 是输入输出信号接口板,共有输入信号 32 点,输出信号 32 点,可以满足一般控制系统的需要。外围检测开关如位置开关和各种显示灯信号全部可以接入 2D-TZ368 接口板中。注意,2D-TZ368 输入输出都是漏型接法,需要提供外部 DC24V 电源。

由于在主回路中有"控制变压器",可以使用"控制变压器"提供的 DC24V 电源。

5. 触摸屏与控制器的连接

触摸屏与控制器直接使用以太网电缆连接。其连接和设置参见第 24 章"触摸屏在机器人上的使用"。

2.9 机器人系统的接地

1. 接地方式

接地是一项很重要的工作,接地不良会导致烧毁机器、伤人或干扰引起的误动作,所以在机器人安装连接时务必接地。

(1) 接地方式有如图 2-17 所示的三种方法,机器人本体及机器人控制器应尽量采用专用接地(图 2-17(a))。

(2) 接地工程应采用 D 种接地(接地电阻 100Ω 以下),以与其他设备分开的专用接地为最佳。

(3) 接地应使用 AWG ♯11(4.2mm^2)以上的电线。接地点应尽量靠近机器人本体、控制器,以缩短接地用电线的距离。

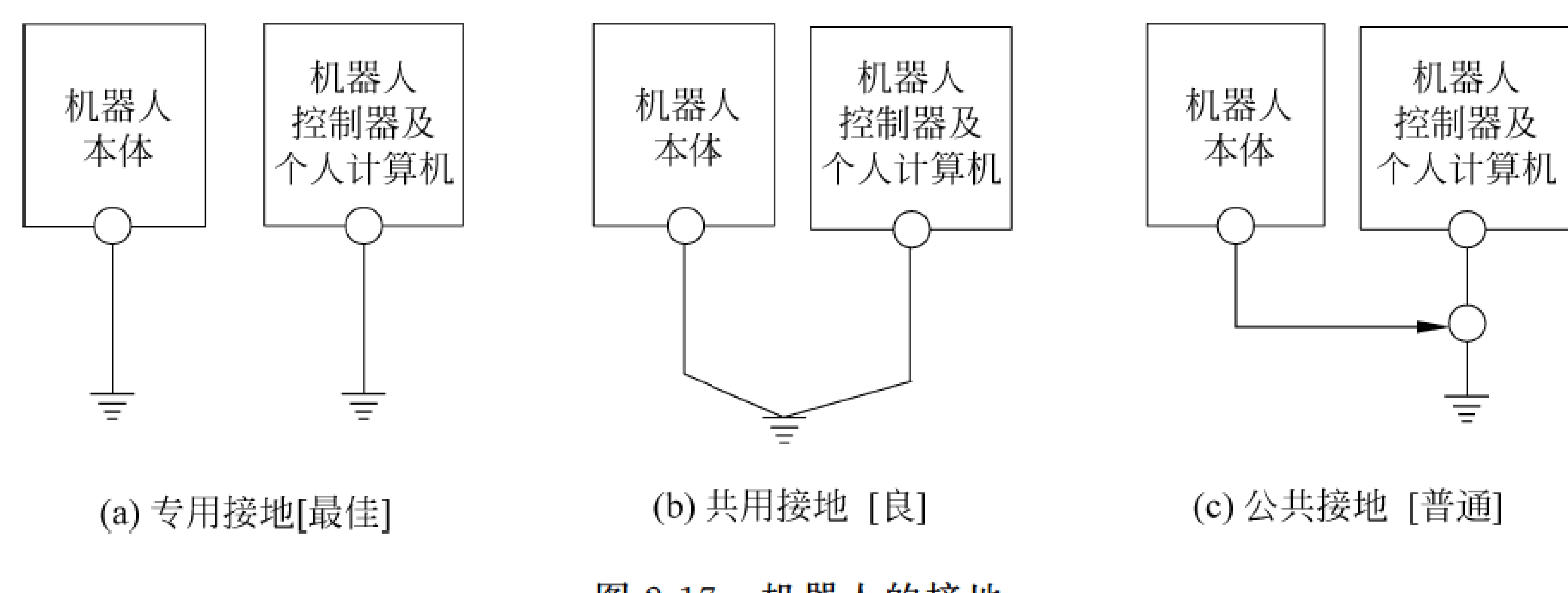


图 2-17 机器人的接地

2. 接地要领

接地线的连接如图 2-18 所示。

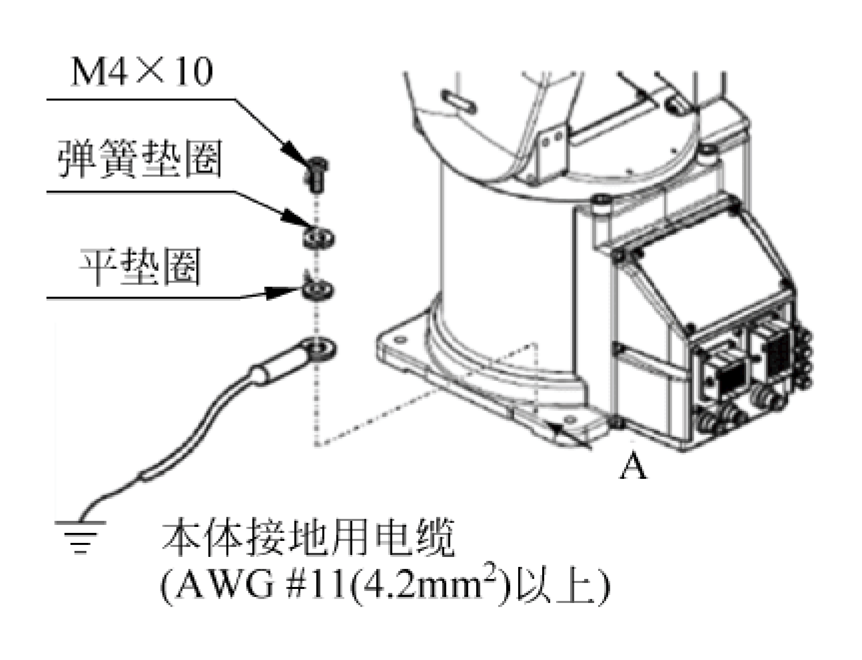


图 2-18 机器人接地的实际接线

接地方法如下。

(1) 准备接地用电缆(AWG ＃11(4.2mm²)以上)及机器人侧的安装螺栓及垫圈。注意不要随意使用面积不够的电线,否则会对机器人系统造成损害。

(2) 接地螺栓部位(A)有锈或油漆的情况下,必须通过锉刀等去除。如果有油漆或锈蚀会引起接地不良,无法消除干扰信号甚至损坏机器。

(3) 将接地电缆连接到接地螺栓部位。

2.10 需要思考的问题

(1) 机器人系统如何连接外部 I/O 信号?

(2) 机器人使用的电源有哪些等级?

(3) 什么是漏型接法?

(4) 什么是源型接法?

(5) 如何连接"工作模式选择"旋钮开关?

(6) 如何连接"急停"开关?

第3章　第3日——使工业机器人动起来

【学习目的】

第3日的学习内容是“让机器人动起来”，通过前两天的学习准备，读者应该已经了解了一些机器人的简单知识，完成了机器人的连接。现在要使用“示教单元”使机器人动起来，破除对机器人的神秘感，学习使用示教单元的基本方法和使机器人运动到指定工作位置点的方法。

3.1　示教单元及其各按键的作用

示教单元也称为“手持式操作单元”，由于常用于操作机器人确认各“位置点”，所以一般称为“示教单元”。

操作单元有许多功能，正确使用操作单元可以起到事半功倍的效果。

学习示教单元上各按键的功能是很有必要的，学习者必须认真学习，多做实验。以下以三菱机器人示教单元R32-TB为例进行说明，如图3-1所示。

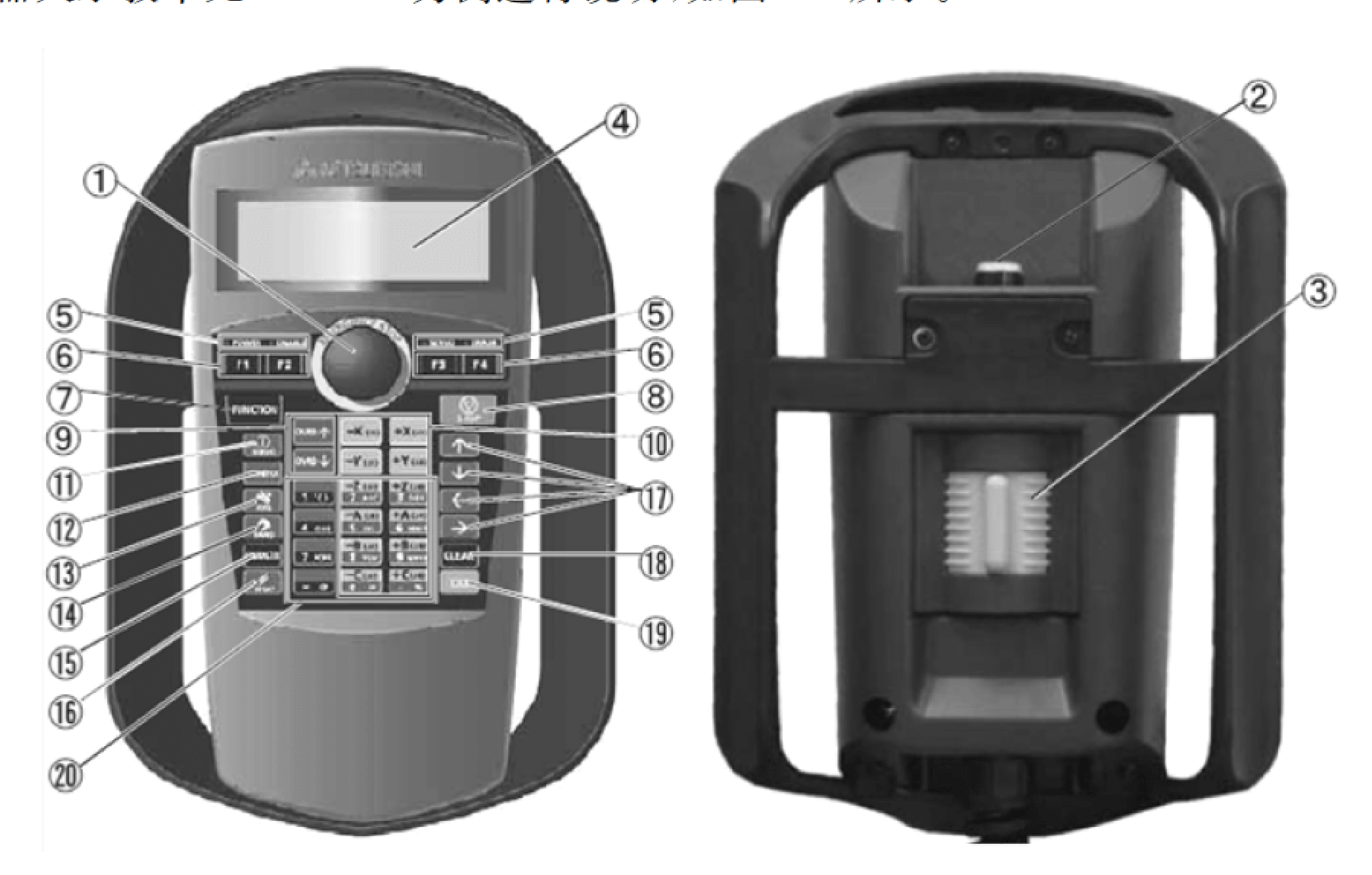


图3-1　示教单元各按键的功能

① EMG. STOP 急停开关

在任何状态下（手动或自动）按下本开关，都可以使机器人进入“急停状态”（伺服＝

OFF),停止一切运动。

② TB ENABLE 使能开关

本开关用于切换示教单元上各按键的有效/无效状态,是一个重要而且常用的开关。按下 TB ENABLE 开关,TB ENABLE 灯亮,表示 TB ENABLE＝ON,示教单元操作有效。同时,示教单元操作有优先权,其他外部设备无法操作机器人。

③ 使能开关

本开关也是使能开关。在“手动模式”下,将本开关拉到中间位置,即可以使伺服＝ON,而本开关在自由位置和“拉到底位置”时,均处于伺服＝OFF 状态,所以也称为“三位置开关”(自由位置、中间位置、拉到底位置),是对示教单元进行多重保护的一个开关。

④ 显示屏

显示屏用于显示相关的数据。

⑤ 显示状态灯

POWER:电源灯,电源 ON 时,POWER＝绿灯。

ENABLE:使能状态灯,示教单元有效时,ENABLE＝绿灯。

SERVO:伺服系统状态灯,伺服系统 ON 时,SERVO＝绿灯。

ERROR:报警状态灯,机器人出现报警时,ERROR＝红灯。

⑥ F1～F4 键

F1～F4 键用于选择显示屏上对应位置的功能,称为功能键。

⑦ FUNCTION 功能键

FUNCTION 功能键用于切换显示屏上的功能菜单。显示屏最下部一次只能显示 4 种功能,如果显示界面的功能多于 4 个,使用 FUNCTION 功能键进行切换。

⑧ STOP 键

STOP 键用于停止正在运行的程序,使运动中的机器人减速停止。

⑨ OVRD＋、OVRD－速度倍率改变按键

OVRD＋、OVRD－速度倍率改变按键用于改变“速度倍率”。

⑩ JOG 操作键

JOG 操作键用于 JOG 运行时指令各轴的运动(X＋,X－,C＋,C－)。

⑪ SERVO 键

SERVO 键用于设置伺服系统 ON/OFF,注意在三位置使能开关＝ON 时才有效。

⑫ MONITOR 键

MONITOR 键为监视模式选择键。MONITOR＝ON 时,示教单元进入监视模式,可监视机器人系统的运动状态。

⑬ JOG 键

JOG 模式选择键。JOG＝ON 时,系统进入 JOG 状态,可以进行各种 JOG 操作。这是最常用的一个按键。

⑭ HAND 键

HAND 键为抓手模式选择键。

⑮ CHARACTER 键

CHARACTER 键为数字/文字切换键,用于切换输入时是数字还是文字。

⑯ RESET 复位键

RESET 复位键用于解除报警状态。同时按下 RESET+EXE 键可执行程序复位。

⑰ 光标键

⑱ CLEAR 删除键

CLEAR 删除键用于删除光标所在位置的内容。

⑲ EXE 执行键

EXE 执行键用于对输入的内容进行确认。

连续按 EXE 键,机器人会动作。

⑳ 数字/文字键

数字/文字键用于输入数字或文字。

3.2 如何使机器人动起来

在学习了解了示教单元上各个按键的作用之后,现在的任务是让机器人动起来,以破除初学者对机器人的神秘感。

让机器人动起来的步骤如下。

(1) 将机器人本体与控制器正常连接。

(2) 将示教单元连接于控制器上。

(3) 暂时不连接其他输入输出信号和操作面板。

(4) 检查安全完毕后上电。

(5) 将 TB ENABLE 开关按下,确认 TB ENABLE 灯亮,这时示教单元为有效状态。

(6) 将三位置使能开关轻拉至中间位置并保持在该位置。

(7) 按下 SERVO 按键,等待 SERVO 绿灯亮。稍后可听见“滴”一声,表示机器人伺服系统=ON。

(8) 选择“速度倍率 OVRD”=10%。

(9) 观察机器人本体的位置,确保机器人动作范围内无障碍物。

(10) 按下 JOG 键,选择 JOG 模式(按键 13)。

注意:现在已经进入 JOG 模式,必须注意安全动作。

(11) 以“点动”方式,逐一按下 J1~J6 键,观察机器人的动作。

如果机器人能够正常运行,就达到了第一阶段的目的。

3.3 学习操作各种 JOG 模式

机器人不同于其他“运动控制器”的特点之一就是,即使是在 JOG 模式下,也有很多类型的 JOG 动作,这取决于采用的不同的“坐标系”。

3.3.1 关节型 JOG

1. 关节型 JOG

如图 3-2 所示以关节轴为对象,以角度为单位实行的“点动操作”就是关节型 JOG,可以

对 J1～J6 轴分别执行 JOG 操作。

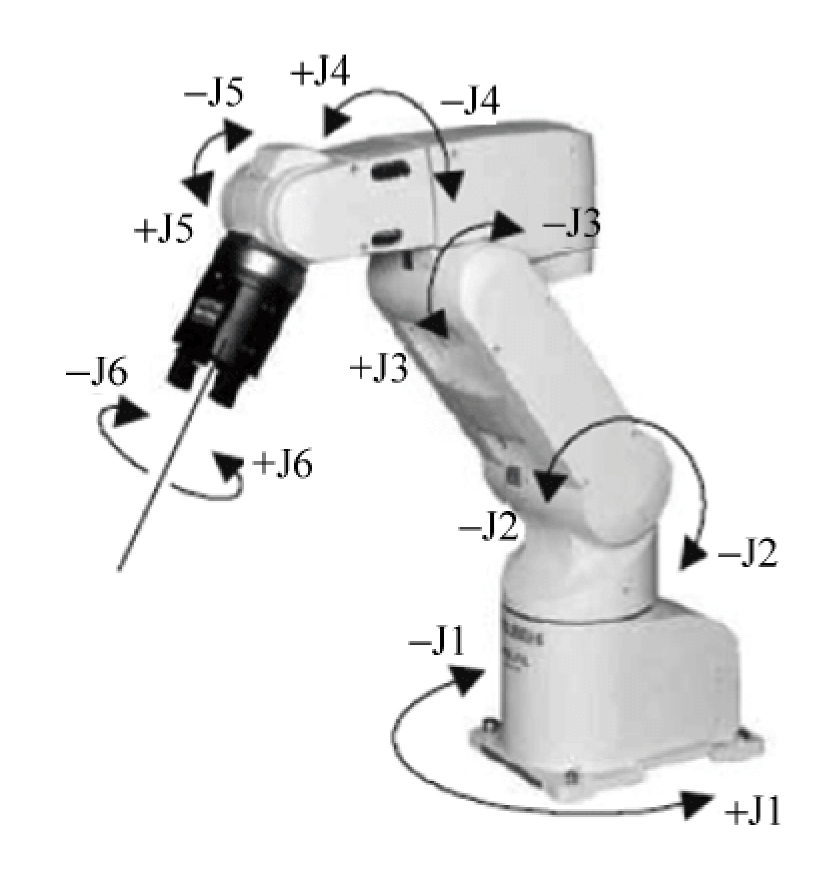


图 3-2　关节型 JOG 示意图

2. 操作步骤

(1) 将 TB ENABLE 开关按下，确认 TB ENABLE 灯亮，这时示教单元为有效状态。

(2) 将“三位置使能开关”轻拉至中间位置并保持在该位置。

(3) 按下 SERVO 按键，等待 SERVO 绿灯亮。稍后可听见“滴”一声，表示机器人伺服系统＝ON。

(4) 按下 JOG 键，选择 JOG 模式(按键 13)。

(5) 根据显示屏上最下排的显示，按下 F1～F4 键，(选择“直交”)。

(6) 以“点动”方式，逐一按下 X、Y、Z、A、B、C 按键，观察机器人的动作。

3.3.2　直交型 JOG

1. 模式

在直交型 JOG 中，以如图 3-3 所示的坐标系为基准，即以“世界坐标系”为基准，机器人控制点在 X/Y/Z 方向上以 mm 为单位运动。而 A/B/C 轴的运动则是旋转运动，以角度为单位。在旋转时，机器人控制点位置不变，抓手的方位改变。

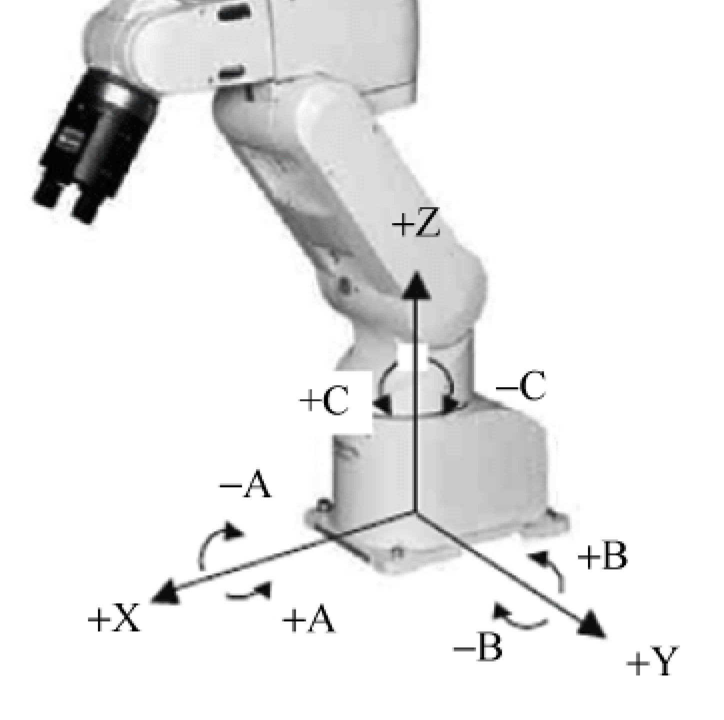


图 3-3　直交型 JOG 示意图

2. 操作步骤

(1) 将 TB ENABLE 开关按下，确认 TB ENABLE 灯亮，这时示教单元为有效状态。

(2) 将“三位置使能开关”轻拉至中间位置并保持在该位置。

(3) 按下 SERVO 按键。等待 SERVO 绿灯亮。稍后可听见“滴”一声，表示机器人伺服系统＝ON。

(4) 按下 JOG 键，选择 JOG 模式(按键 13)。

(5) 根据显示屏上最下排的显示，按下 F1～F4 键(选择“直交”)。

(6) 以“点动”方式，逐一按下 X、Y、Z、A、B、C 按键，观察机器人的动作。

3.3.3　TOOL 型 JOG

1. 模式

TOOL 型 JOG 就是以“TOOL 坐标系”为基准进行的 JOG 运行，如图 3-4 所示。

这种 TOOL 型 JOG 以“TOOL 坐标系”为基准，在 TOOL 坐标系的 X/Y/Z 方向做直线运动，单位为 mm。在 A/B/C 轴方向做旋转运动，以角度为单位。

TOOL 型 JOG 与直交型 JOG 的不同就是依据的坐标系不同，所以使用时要预先设置 MEXTL 参数，也就是预先设置 TOOL 坐标系。

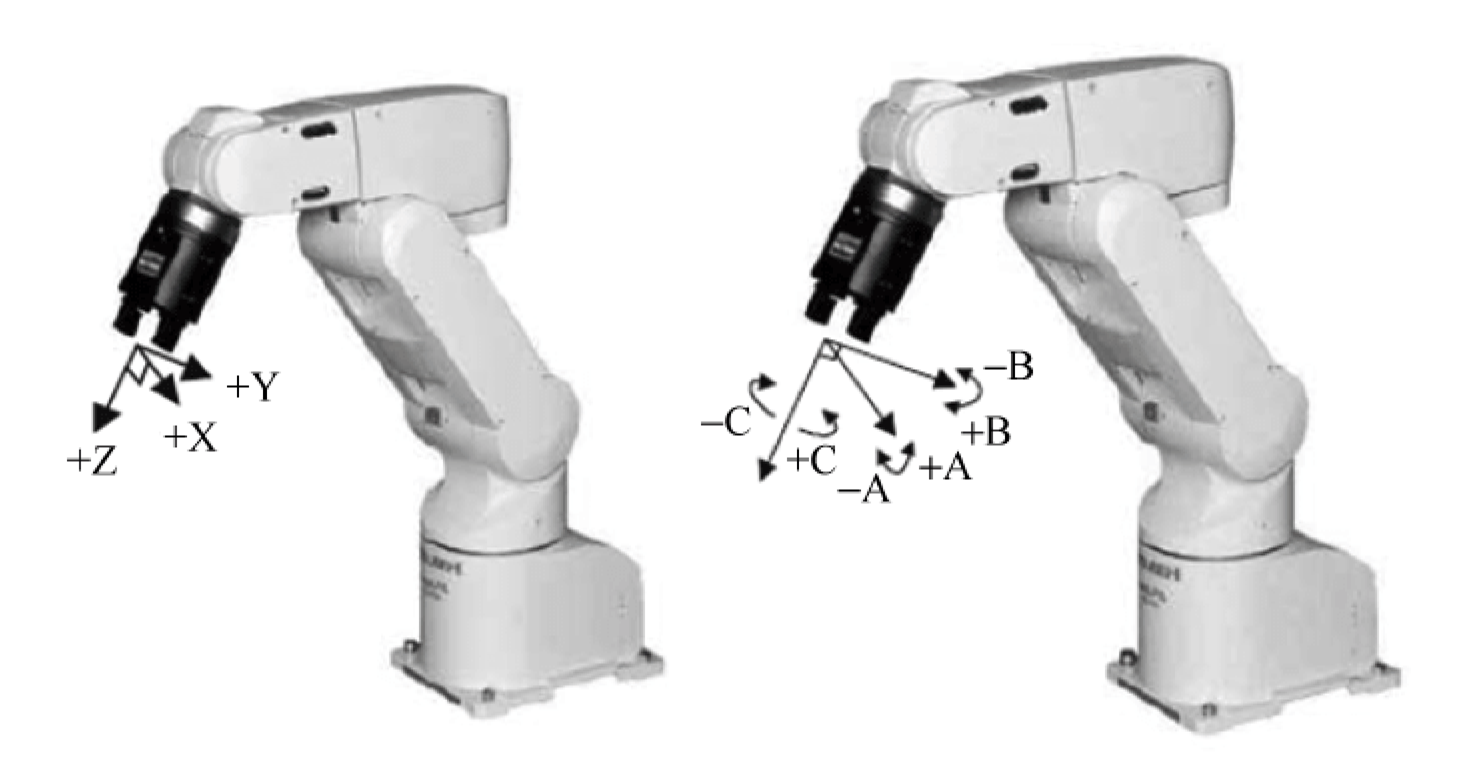


图 3-4 TOOL 型坐标系示意图

2. 操作步骤

(1) 将 TB ENABLE 开关按下，确认 TB ENABLE 灯亮，这时示教单元为有效状态。

(2) 将“三位置使能开关”轻拉至中间位置并保持在该位置。

(3) 按下 SERVO 按键，等待 SERVO 绿灯亮。稍后可听见“滴”一声，表示机器人伺服系统＝ON。

(4) 按下 JOG 键，选择 JOG 模式(按键 13)。

(5) 根据显示屏上最下排的显示，按下 F1～F4 键(选择 TOOL)。

(6) 以“点动”方式，逐一按下 X、Y、Z、A、B、C 按键，观察机器人的动作。

3.3.4 三轴直交型 JOG

1. 模式

三轴直交型 JOG 在 X/Y/Z 方向上是以“世界坐标系”为基准，移动单位是 mm。但是 A/B/C 三轴的移动则是对应 J4/J5/J6 轴，以角度为单位，如图 3-5 所示。这种方式综合了两种坐标系的优势。

2. 操作步骤

(1) 将 TB ENABLE 开关按下，确认 TB ENABLE 灯亮，这时示教单元为有效状态。

(2) 将“三位置使能开关”轻拉至中间位置并保持在该位置。

(3) 按下 SERVO 按键，等待 SERVO 绿灯亮。稍后可听见“滴”一声，表示机器人伺服系统＝ON。

(4) 按下 JOG 键，选择 JOG 模式(按键 13)。

(5) 根据显示屏上最下排的显示，按下 F1～F4 键(选择“三轴直交”)。

(6) 以“点动”方式，逐一按下 X、Y、Z、A、B、C 按键，观察机器人的动作。

图 3-5 三轴直交型 JOG 示意图

3.3.5 圆筒型 JOG

1. 模式

圆筒型 JOG 首先要建立一个圆筒型坐标系。如图 3-6 所示，在圆筒型坐标系中，X 坐

标表示圆筒的半径，Z 坐标表示圆筒的高度，Y 坐标表示圆筒的旋转角度（也就是 J1 轴的角度）。其余 A/B/C 轴的旋转方向如图 3-6 所示。这样圆筒型 JOG 就相当于机器人控制点在一个圆筒壁上做运动。或者说，如果是一个圆筒壁上的运动，就选取圆筒型 JOG 最为适宜。

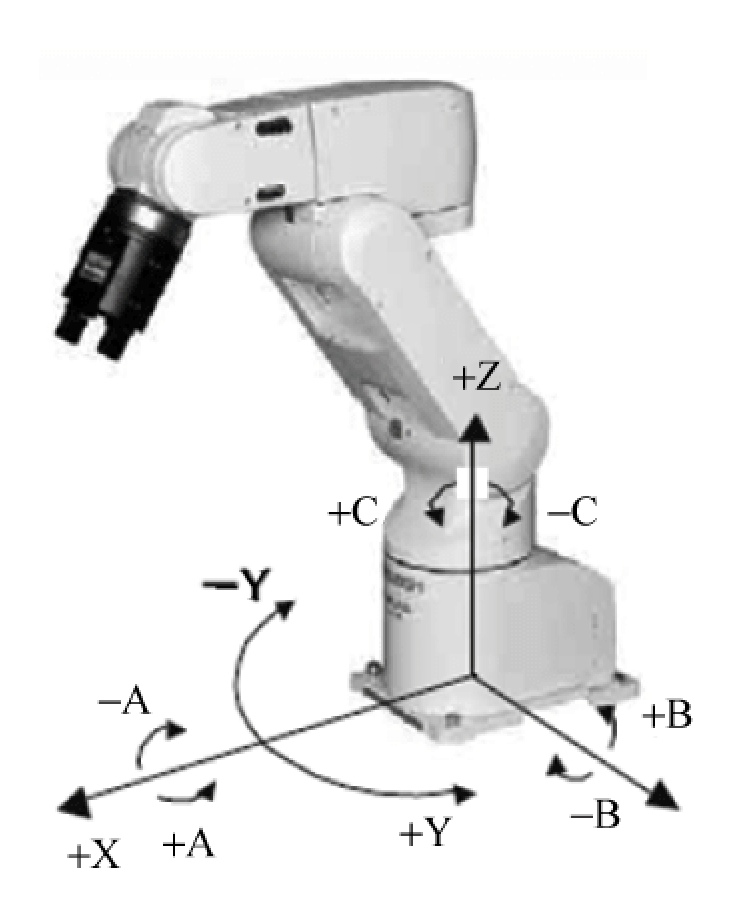


图 3-6　圆筒型坐标系

2. 操作步骤

（1）将 TB ENABLE 开关按下，确认 TB ENABLE 灯亮。这时示教单元为有效状态。

（2）将"三位置使能开关"轻拉至中间位置并保持在该位置。

（3）按下 SERVO 按键，等待 SERVO 绿灯亮。稍后可听见"滴"一声，表示机器人伺服系统＝ON。

（4）按下 JOG 键，选择 JOG 模式（按键 13）。

（5）根据显示屏上最下排的显示，按下 F1～F4 键（选择"圆筒 JOG"）。

（6）以"点动"方式，逐一按下 X、Y、Z、A、B、C 按键，观察机器人的动作。

3.3.6　工件型 JOG

1. 模式

工件型 JOG 就是以工件坐标系进行的点动操作。事实上，如果要做轨迹型的运动，工件的图纸是已经设计完毕的。工件的安装与机器人的相对位置也是固定的。工件坐标系如图 3-7 所示，所以工件 JOG 就是沿着工件坐标系进行的 JOG 运动，与直交型 JOG 相同，只是坐标系位置不同。

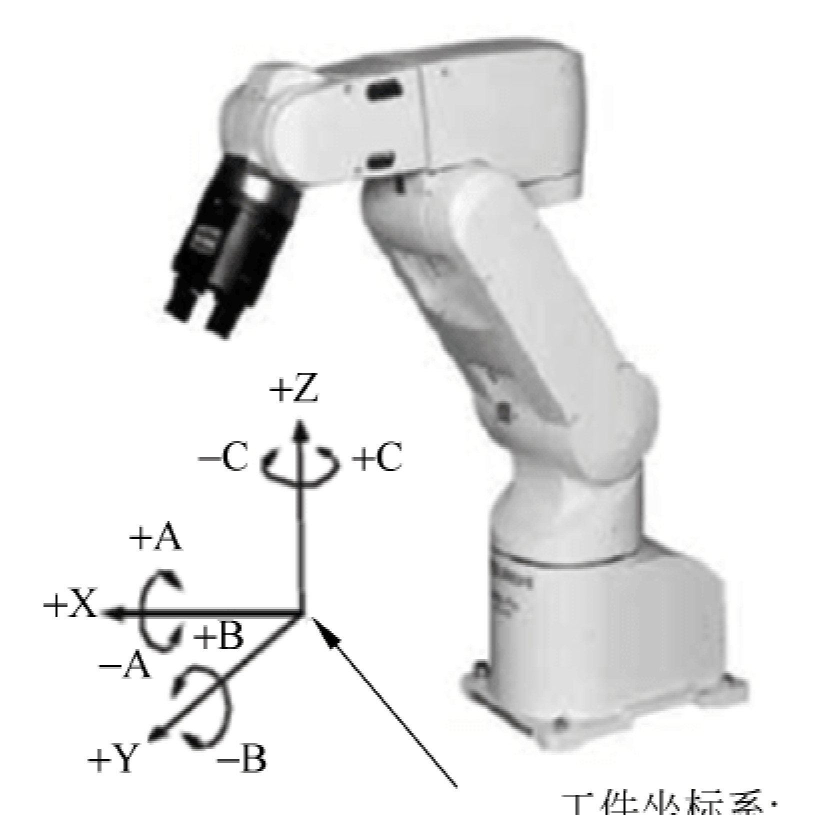


图 3-7　工件型 JOG 示意图

机器人控制点在 X/Y/Z 方向上以 mm 为单位运动。而 A/B/C 轴的运动则是旋转运动，以角度为单位。

2. 操作步骤

（1）将 TB ENABLE 开关按下，确认 TB ENABLE 灯亮，这时示教单元为有效状态。

（2）将"三位置使能开关"轻拉至中间位置并保持在该位置。

（3）按下 SERVO 按键，等待 SERVO 绿灯亮，稍后可听见"滴"一声，表示机器人伺服系统＝ON。

（4）按下 JOG 键，选择 JOG 模式（按键 13）。

（5）根据显示屏上最下排的显示，按下 F1～F4 键（选择"工件 JOG"）。

（6）以"点动"方式，逐一按下 X、Y、Z、A、B、C 按键，观察机器人的动作。

3.4　需要思考的问题

（1）使用示教单元进行 JOG 运行的基本步骤有哪些？
（2）什么是关节型 JOG？
（3）什么是直交型 JOG？

第4章 第4日——使用示教单元进行编程、调试及设置参数

【学习目的】

第4日的学习内容是对示教单元功能的深度学习，要充分理解“示教单元”的丰富功能。即使在没有计算机的情况下，只要有“示教单元”就可以进行编程、调试，设置参数，驱动机器人正常运行。

4.1 示教单元的“整列”功能

1. 功能

“整列”功能就是使机器人的抓手就近回到距离“当前位置”最靠近的90°方向，如图4-1所示。在实际应用中，如果需要抓手迅速对准工件，这是一种快捷的方法。

(1) 如果没有设置“TOOL坐标系”，经过“整列”后，抓手就到达图中的前一个位置。

(2) 如果设置了“TOOL坐标系”，经过“整列”后，抓手就到达图中的后一个位置。

可以看到是以“控制点”为基准，“控制点”位置不变，抓手的位置发生改变。

2. 操作步骤

(1) 选择“手动模式”。

(2) 将TB ENABLE开关按下，确认TB ENABLE灯亮，这时示教单元为有效状态。

(3) 将“三位置使能开关”轻拉至中间位置并保持在该位置。

(4) 按下SERVO按键，等待SERVO绿灯亮。稍后可听见“滴”一声，表示机器人伺服系统=ON。

(5) 按下HAND键，显示“抓手”界面如图4-2所示。

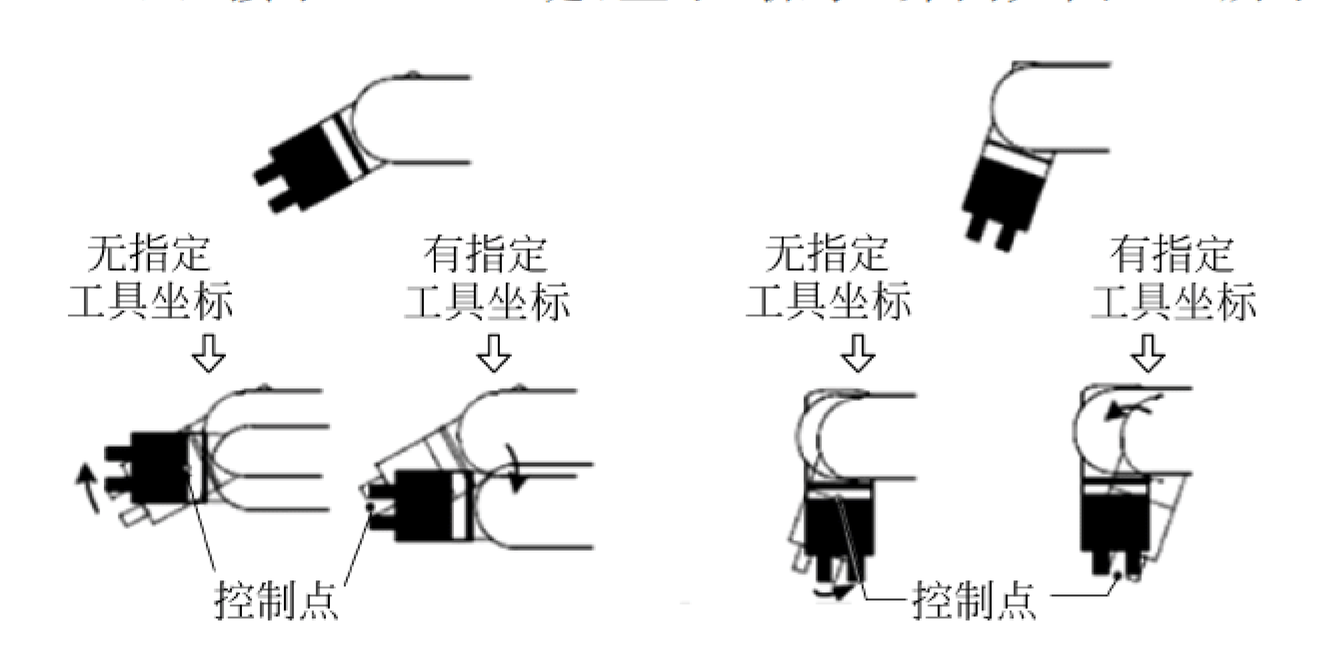


图4-1　整列位置示意图

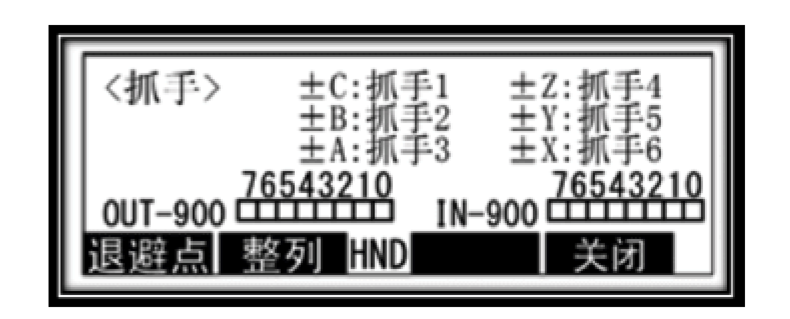


图4-2　“抓手”界面

(6) 按下并保持(按下)对应“整列”功能的 F2 键,机器人动作,执行“整列”动作。

4.2 程序编辑

示教单元的重要功能之一是可以编辑程序,不使用“编程软件”也可以完成程序编辑的相关任务。相关的指令可在后续章节学习。

以下是编辑程序的步骤。

(1) 上电后,按 EXE 键进入菜单界面,选择进入“管理/编辑”界面。

(2) 按下对应“新建”功能的 F3 键,进入如图 4-3 所示界面。

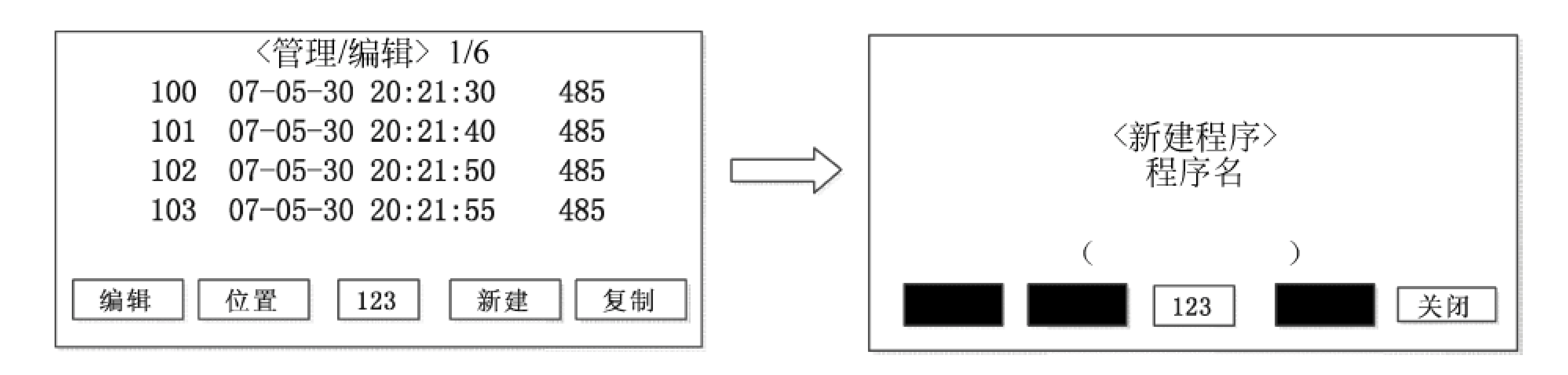


图 4-3 新建一个程序

(3) 输入程序名,按 EXE 键,进入如图 4-4 所示指令编辑界面。

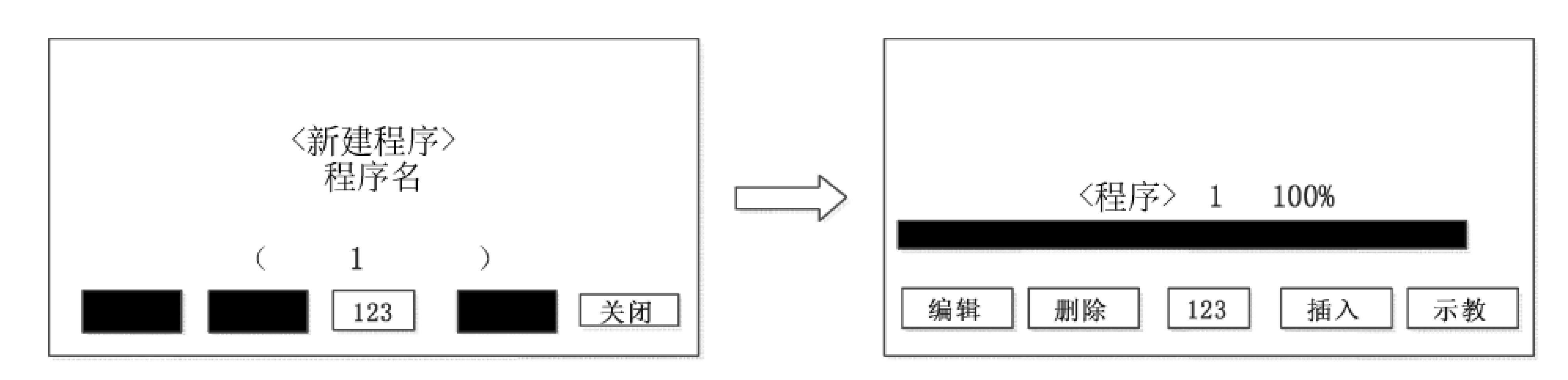


图 4-4 对新程序命名

(4) 进入指令编辑。以输入以下程序为例:

```
1 Mov P1
2 Mov P2
3 End
```

① 按下对应“插入”功能的 F3 键,进入“插入编辑”状态,如图 4-5 所示。

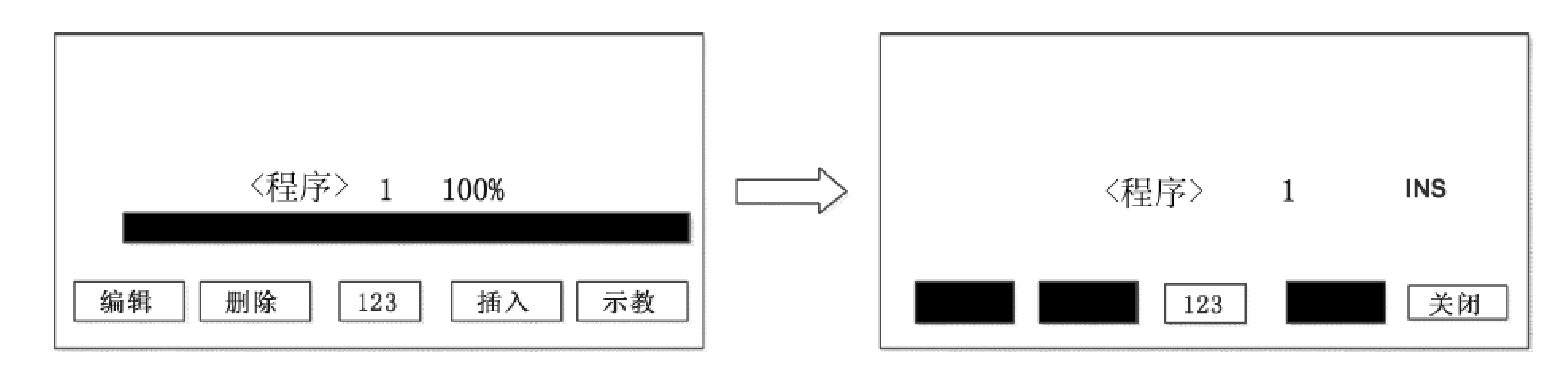


图 4-5 程序输入编辑界面

② 输入“步序号”；按 CHARACTER 键，选择数字输入，输入数字“1”，进入如图 4-6 所示界面。

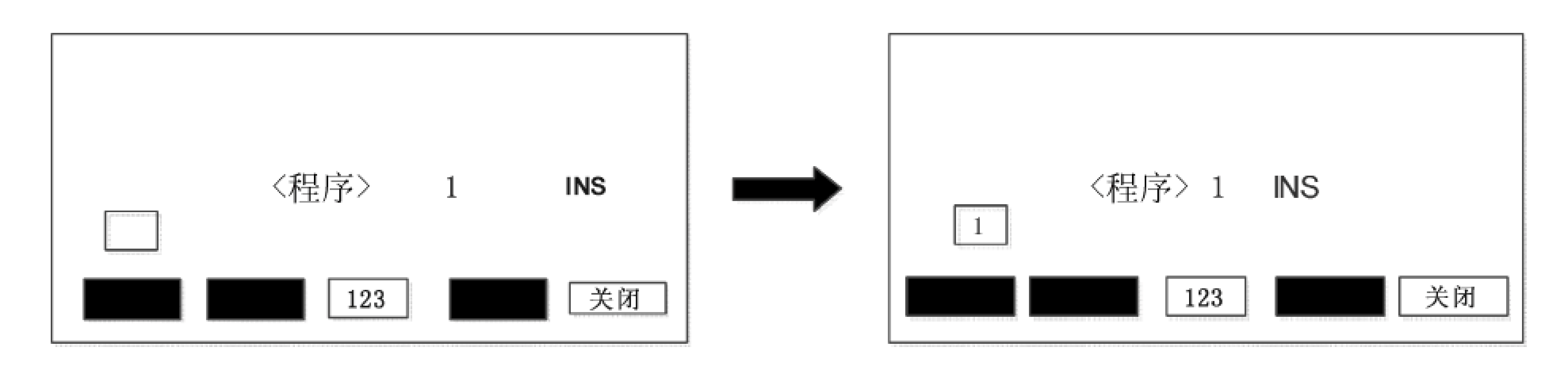


图 4-6 程序步序号输入编辑界面

③ 输入“MOV P1”指令。注意在“MOV”和“P1”之间有空格，如图 4-7 所示。

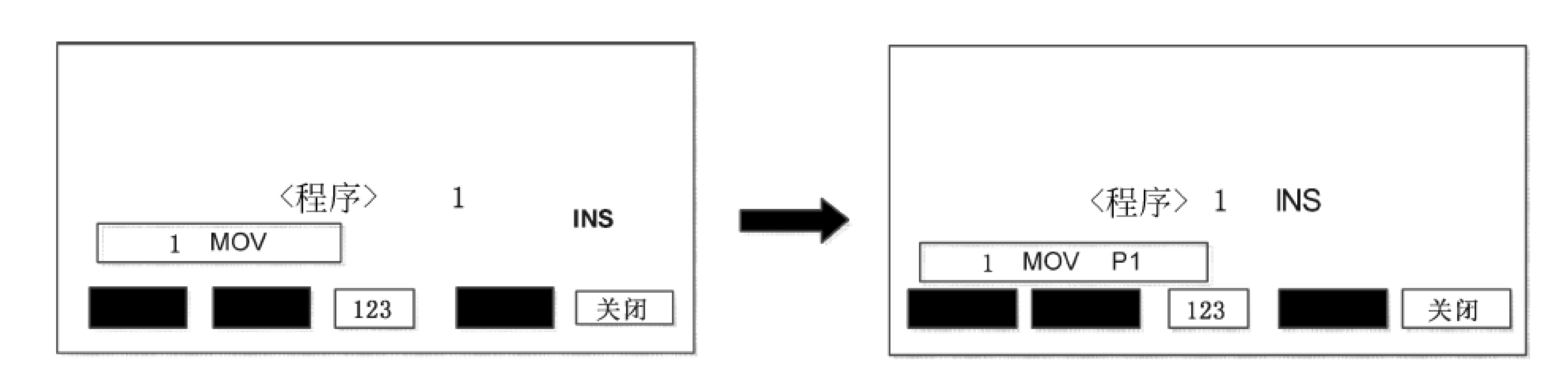


图 4-7 程序输入界面

④ 输入程序的确认。

在输入了程序指令后，还要进行“确认”操作，即对输入的程序指令进行确认。操作如下。

选定程序行(例如选择 1 MOV P1)→按 EXE 键。

如图 4-8 所示光标移到下一行。同样的操作，可以对所有的程序行进行确认。

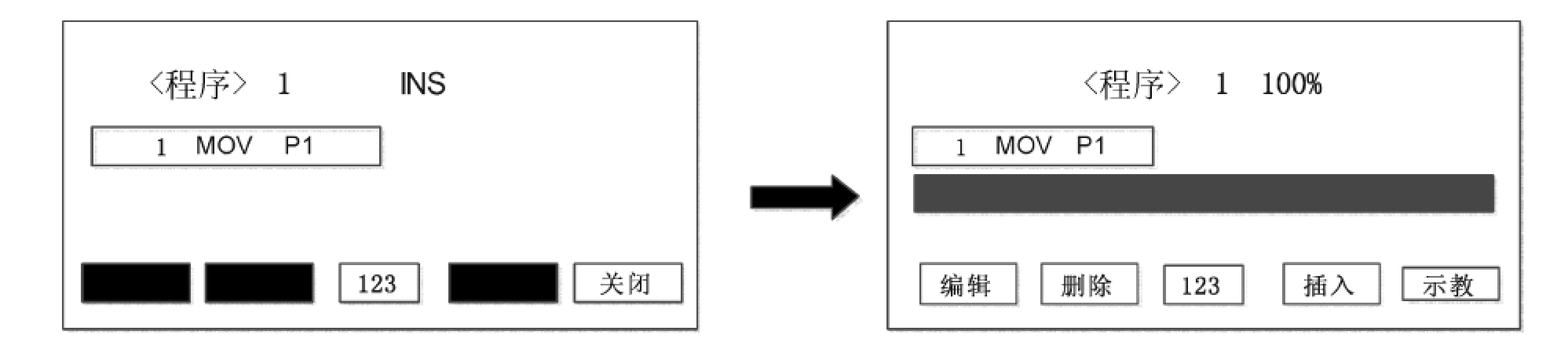


图 4-8 程序输入确认界面

⑤ 编制程序结束。

在编制程序结束后，按“关闭”键，回到上一级菜单“管理/编辑”，如图 4-9 所示。

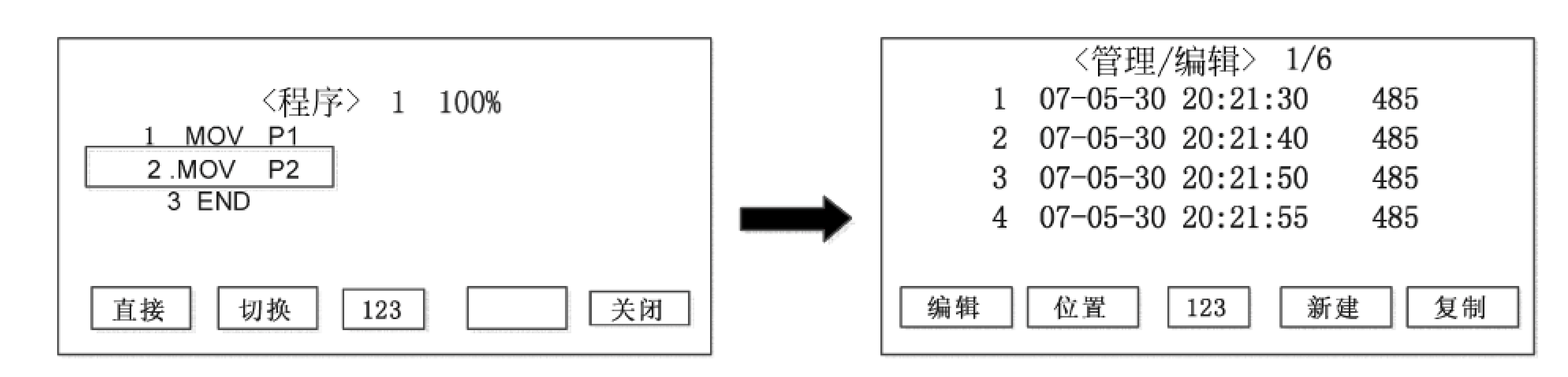


图 4-9 回到上一级菜单“管理/编辑”

4.3 程 序 修 正

程序修正是经常性的工作，程序修正方法步骤如下。

例如，将“MOV P5”修改为“MVS P5”。

（1）进入需要修改的程序界面。

（2）选择需要修改的程序行（5 MOV P5）。

（3）按下“编辑”键，进入“修改”界面，如图 4-10 所示。

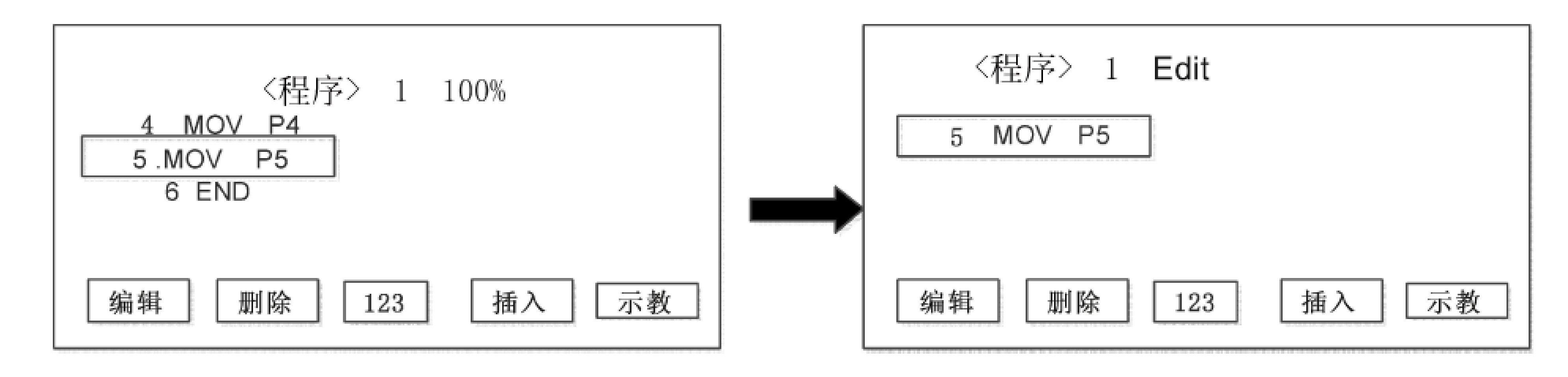


图 4-10　程序修改界面

（4）输入 MVS 指令，再按下 EXE 键，确认修改完成，如图 4-11 所示。

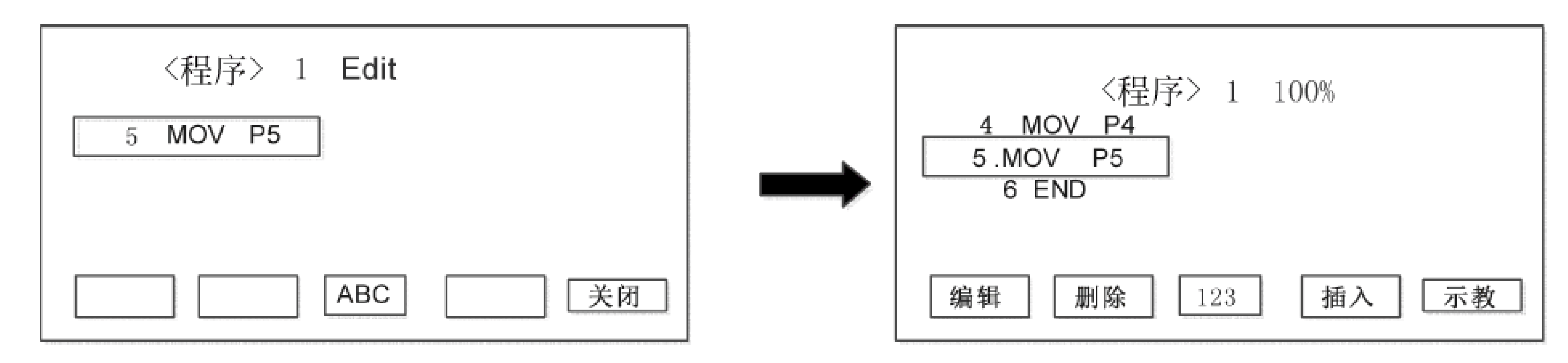


图 4-11　程序修改完成界面

（5）按“关闭”键，保存修改的程序。

4.4 示 教 操 作

本操作就是要将“当前位置点”设置为自动程序中的一个“工作点”。这是示教单元最重要的工作之一。在实际工作中，可以观察到机器人的工作点位置（例如 P1 点或 P2 点），但具体的坐标数值不知道。利用 JOG 功能，将机器人直接移动到 P1 点，在显示屏上就可以读出 P1 点的数据，同时就可以将“当前点”设置为程序中的 P1 点。这个过程就是“示教”。

示教操作如下（以 P5 点为例）。

（1）在“指令编辑”界面进行的示教操作。

① 使用 JOG 功能，将机器人直接移动到（预定的）P5 点，并在显示屏上观察“当前点”的数据。

② 在“指令编辑”界面选择程序行“5 MOV P5”。

③ 选择按下对应“示教”功能的按键 F4，进入“示教确认”界面，如图 4-12 所示。

④ 按下“是”按键，“当前位置点”被设置为 P5 点。同时返回“上一级程序编辑”界面，如图 4-13 所示。

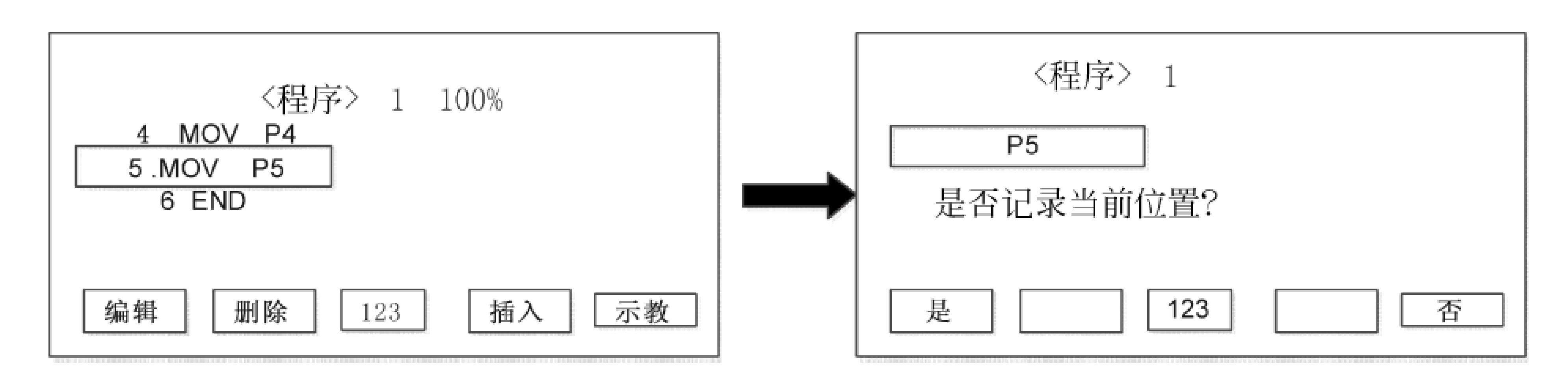


图 4-12 进入"示教确认"界面

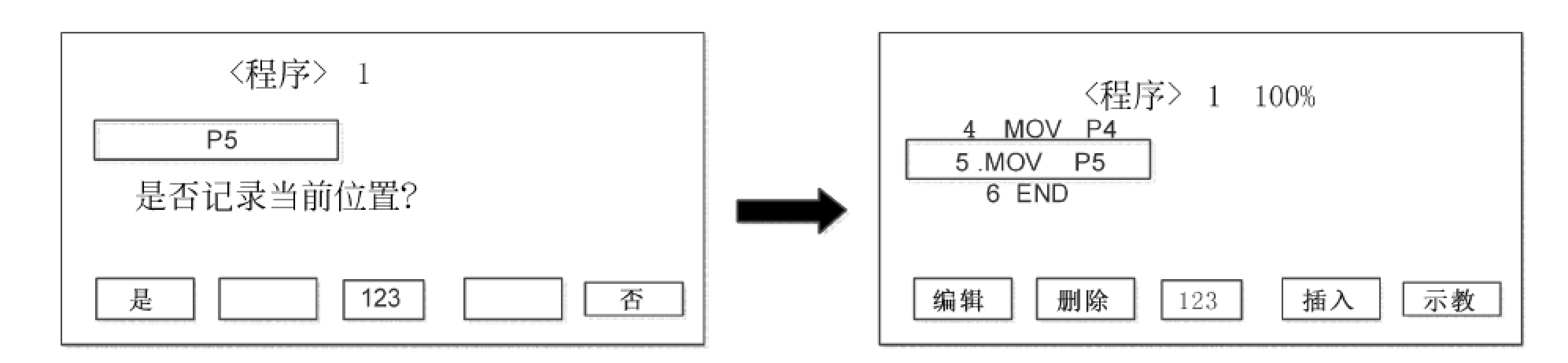


图 4-13 示教点确认

⑤ 示教操作完成。

(2) 在"位置编辑"界面进行的示教操作。

"位置编辑"界面是专门对各"工作位置点"进行编辑的界面。可以在"位置编辑"界面调出各"工作位置点",利用示教功能进行设置。

操作步骤如下。

① 使用JOG功能,将机器人直接移动到(预定的)P5点,并在显示屏上观察"当前点"的数据。

② 在"指令编辑"界面按下切换键进入"位置编辑"界面,如图4-14所示。

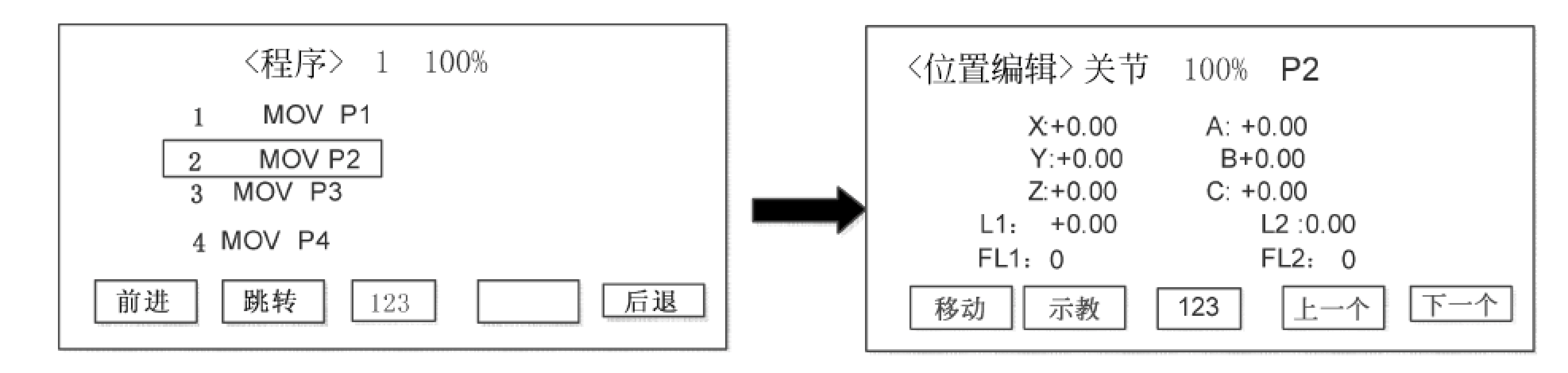


图 4-14 进入"位置编辑"界面

③ 使用"上一个""下一个"按键调出P5点,如图4-15所示。

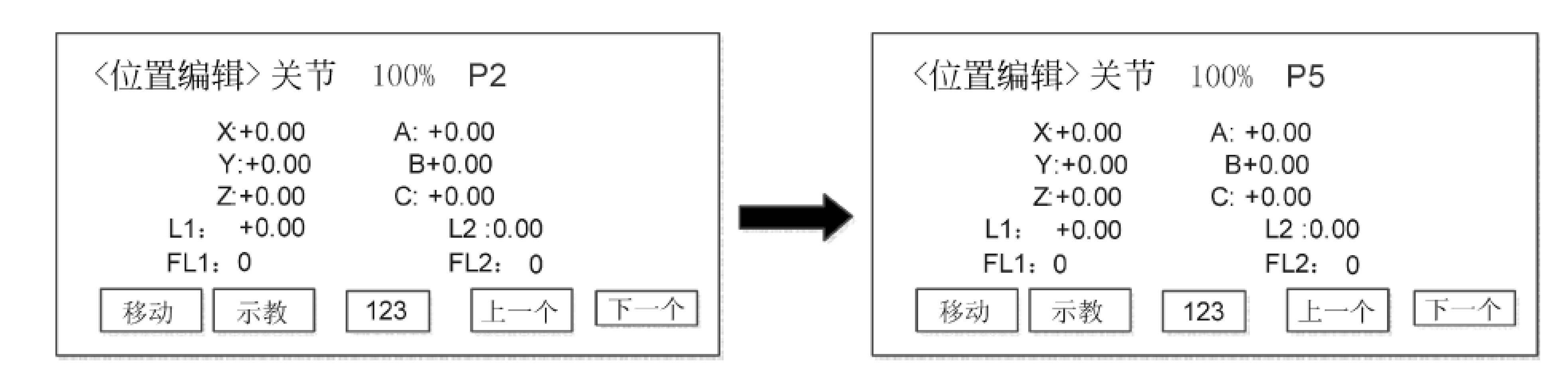


图 4-15 调出P5点

④ 按下对应“示教”功能的按键 F2，进入“示教确认”界面，如图 4-16 所示。

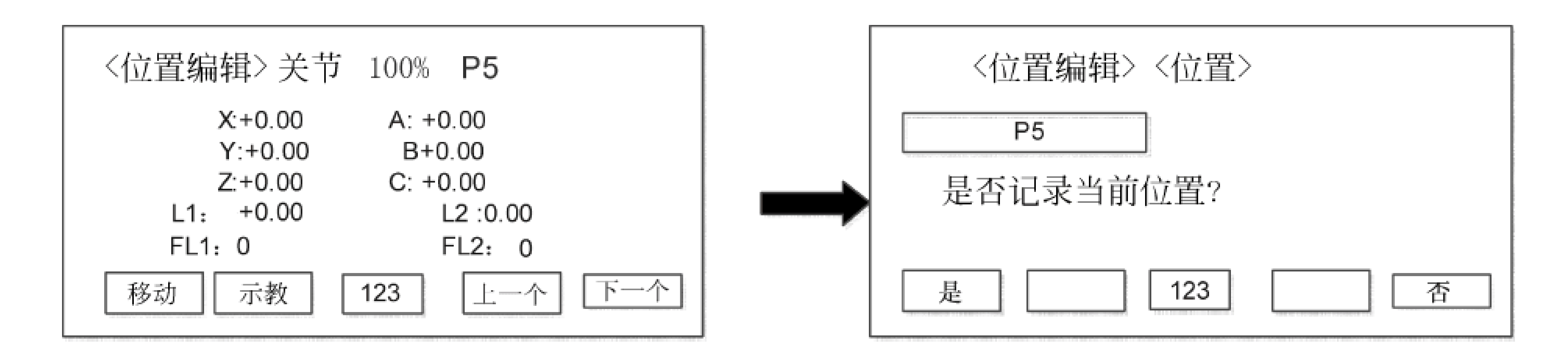


图 4-16 进入“示教确认”界面

⑤ 按下“是”按键，“当前位置点”被设置为 P5 点，如图 4-17 所示。

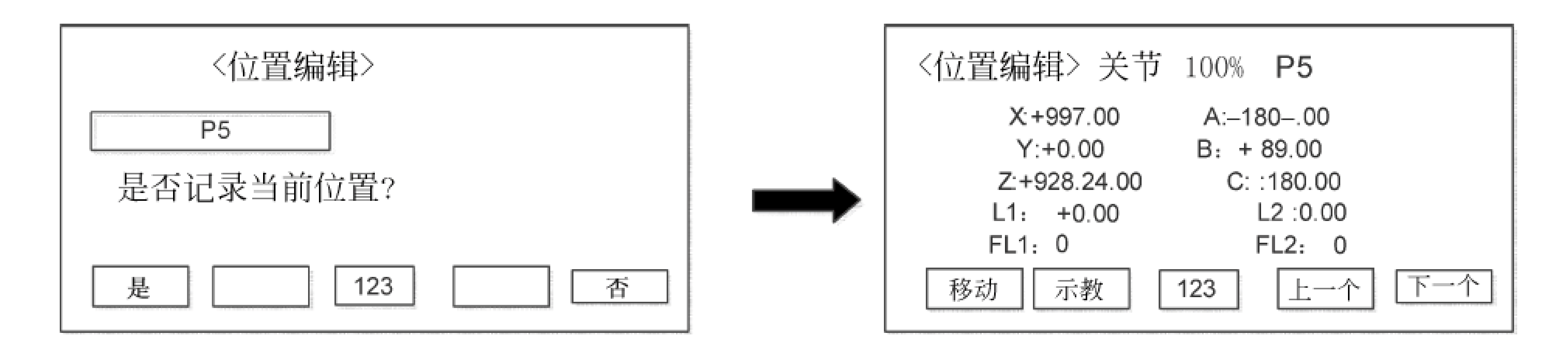


图 4-17 设置 P5 点

⑥ 示教操作完成。

4.5 向预先确定的位置移动

在实际现场，可能出现要求机器人往一个“确定的点”运动。例如，对一个点进行反复的调整。

使用“示教单元”可以实现这一要求。操作步骤如下。

(1) 伺服 ON。

(2) 进入“位置编辑”界面，调出预想位置点，图中为 P1 点，如图 4-18 所示。

(3) 按住对应“移动”功能的 F1 键，直到机器人移动到该位置 P1 点。

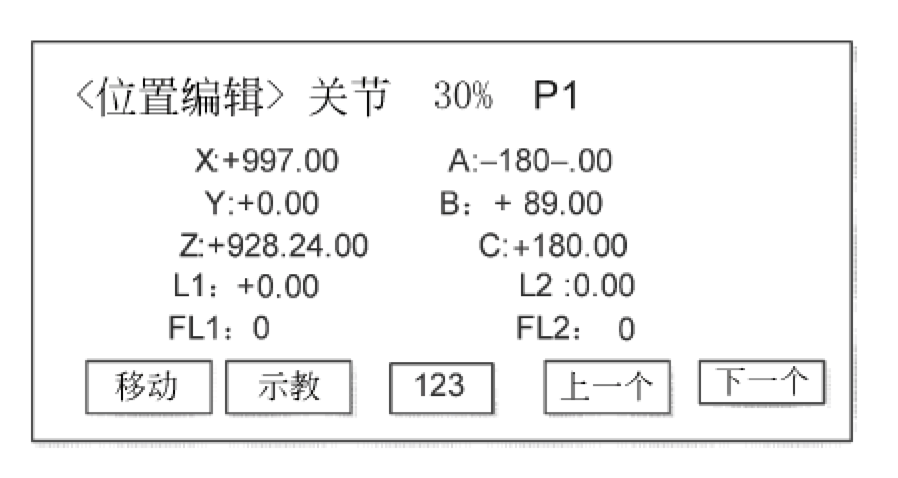


图 4-18 调出预想位置点

4.6 位置数据的 MDI(手动数据输入)

如果需要调整“位置点的某一轴数据”，可以使用 MDI 功能。MDI 即“手动数据输入”。操作如下(修改 P50 点数据)。

(1) 进入“位置编辑”界面，调出 P50 点，如图 4-19 所示。

(2) 在 Z 位置输入“50”后按 EXE 键，如图 4-20 所示。

(3) 修改完成。

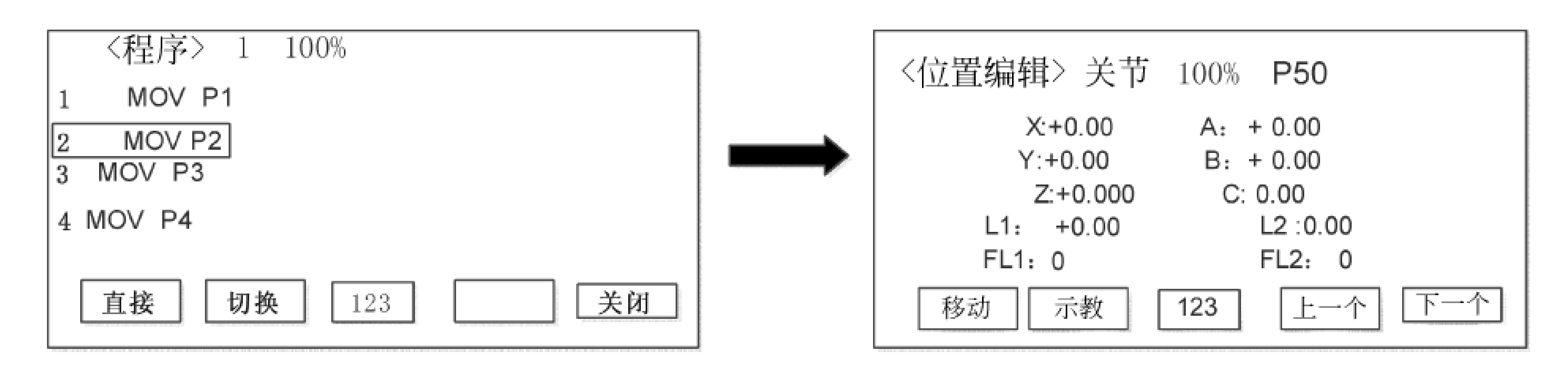


图 4-19 调出 P50 点

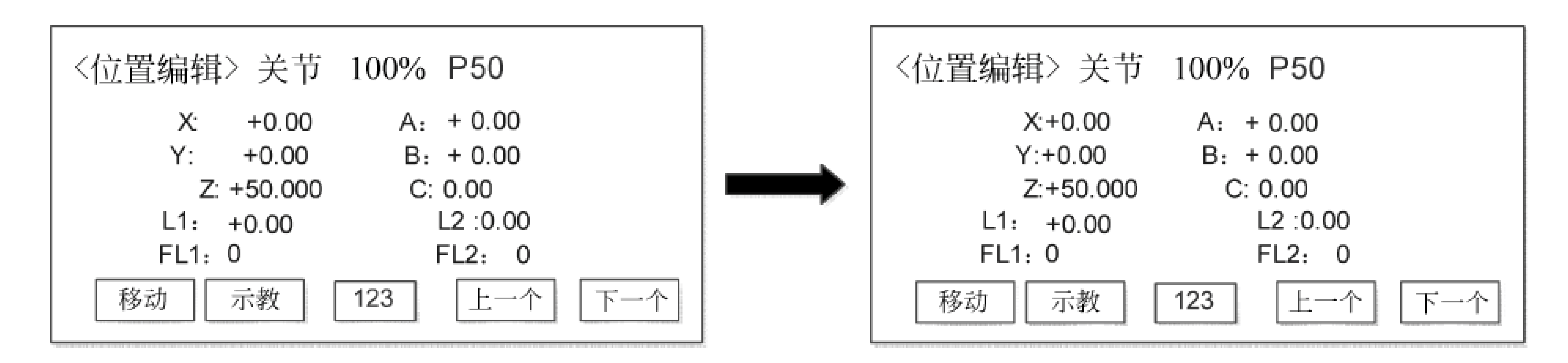


图 4-20 Z 位置输入“50”

4.7 调试功能

使用示教单元也可以进行程序调试。

4.7.1 单步运行

单步运行是程序调试中最常用的功能。操作步骤如下。

(1) 按“菜单”→“管理/编辑”→“编辑”键，进入“程序”编辑界面，如图 4-21 所示。

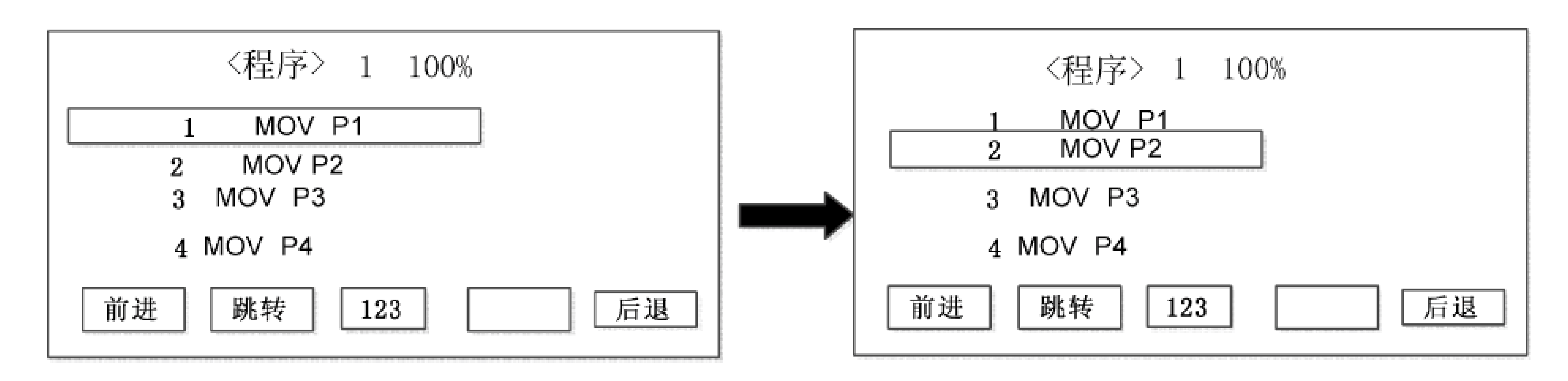


图 4-21 进入“程序”编辑界面

(2) 按下对应“前进”功能的 F1 键，系统执行“光标所在的程序行”后，停止。

(3) 每按一次“前进”键，单步执行一个“程序行”。

4.7.2 程序逆向运行

一般程序运行是按程序行号顺向运行的。在下面的程序中，程序运行的顺序是 P1-P2-P3-P4。

```
1  MOV  P1
2  MOV  P2
3  MOV  P3
4  MOV  P4
```

而程序逆向运行则是运行到 P4 点后，再一步一步按 P4-P3-P2-P1 顺序逆向运行，如图 4-22 所示。

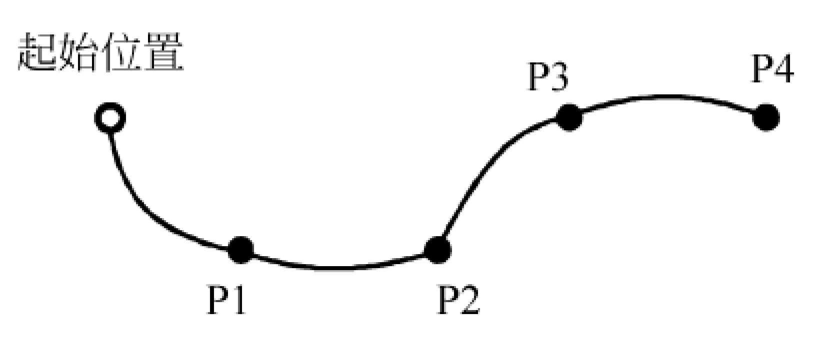


图 4-22 逆向运行示意图

操作步骤如下。

程序当前行已经是“4 MOV P4”，到达了 P4 点。

(1) 进入“程序”编辑界面，如图 4-23 所示。

(2) 按下对应“后退”功能的 F4 键，直到程序执行到第 3 行“3 MOV P3”。

(3) 每按下对应“后退”功能的 F4 键，程序后退一行。

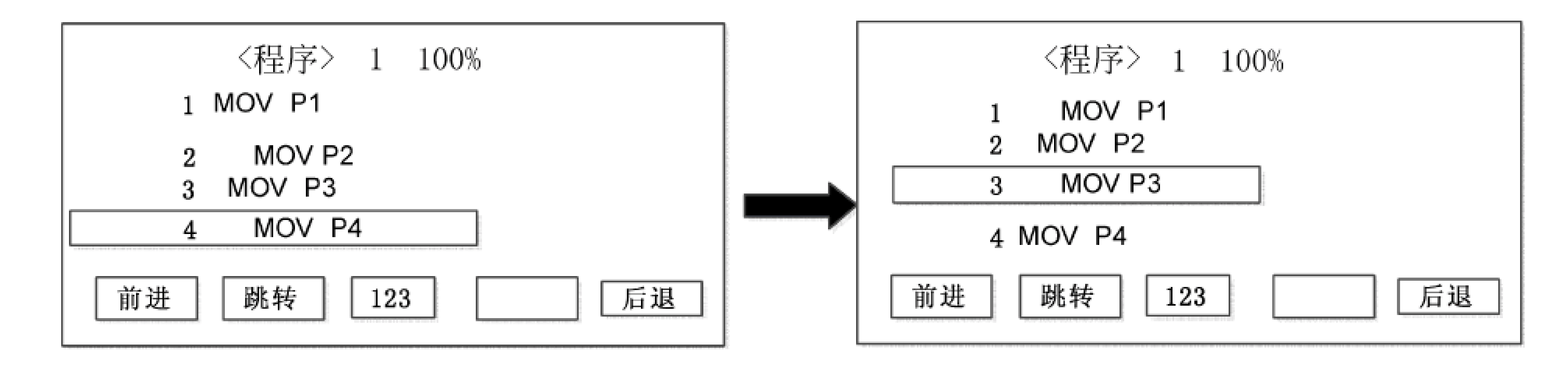


图 4-23 进入“程序”编辑界面

4.7.3 跳转

如果希望执行某一指定的程序行，可以直接从当前行转到指定行，这就是跳转功能。操作如下。

(1) 进入“程序”编辑界面，如图 4-24 所示。

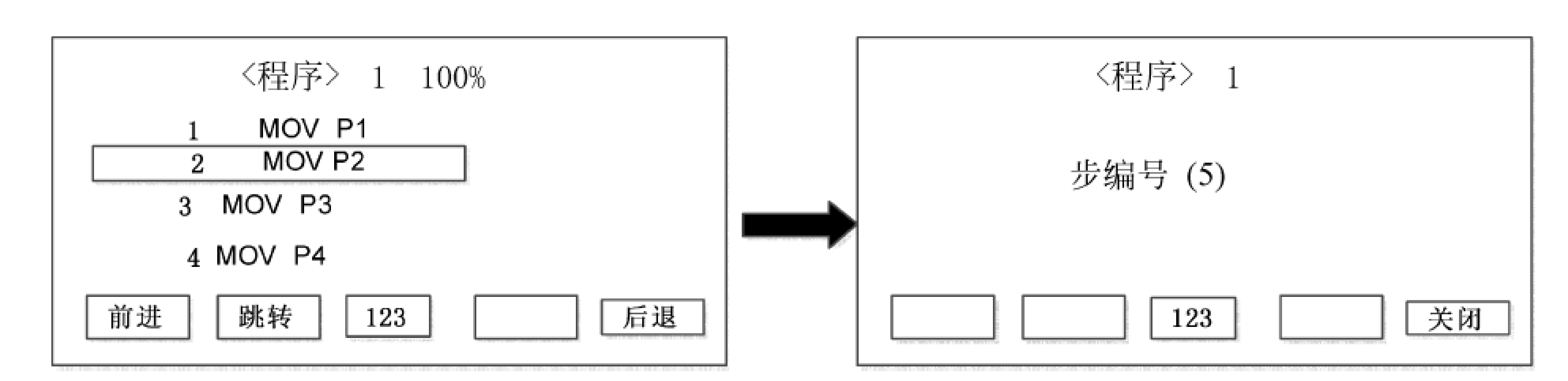


图 4-24 进入“程序”编辑界面

(2) 按下对应跳转功能的 F2 键，进入跳转程序行号确认界面，如图 4-25 所示。

(3) 设置希望跳转的程序行号(例如：程序行号=5)。

(4) 光标跳到第 5 步程序行。

(5) 按下“前进”功能键，程序单步执行。

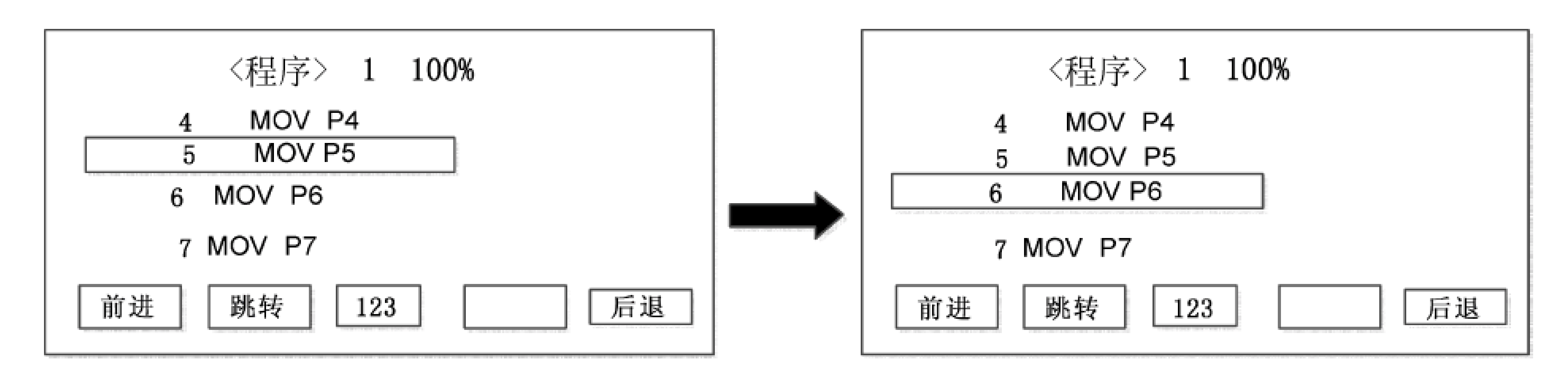


图 4-25 按下对应“跳转”功能的 F2 键

4.8 示教单元的高级编程管理功能

示教单元作为一个完善的“手持操作器”，具备丰富的操作功能，本节将全部介绍。在第4日的学习中，不必全部阅读。可以在后面学习了相关的程序指令及参数等内容之后，再来参考本节的学习。

4.8.1 管理和编辑（程序）

本功能是建立新的运动程序或对原有的程序进行编辑，具有下列功能。

（1）建立一个新的程序；

（2）编辑原有程序；

（3）编辑“位置点”；

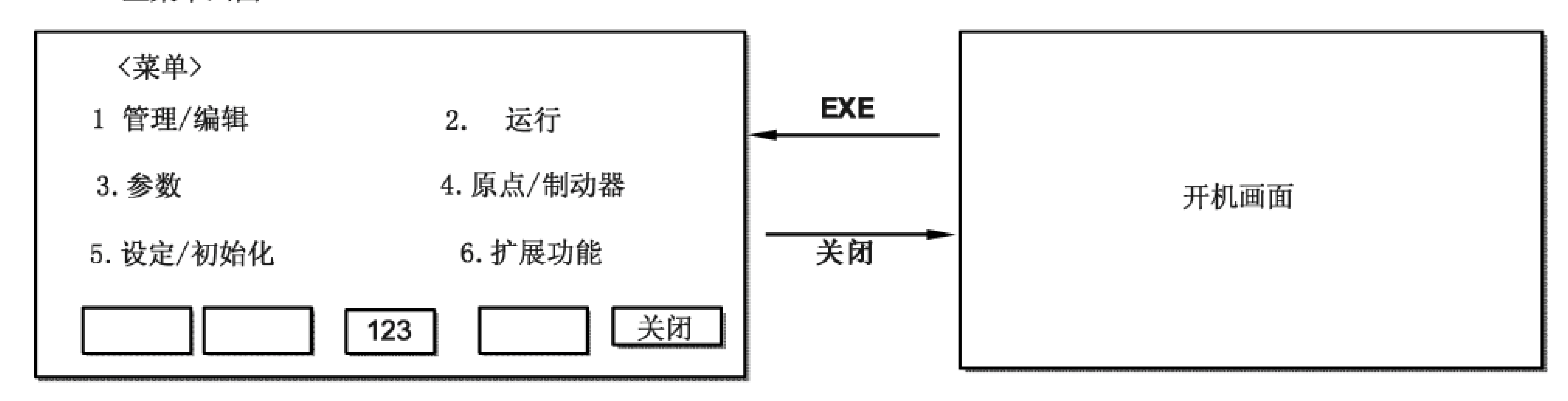
（4）复制程序；

（5）重新命名程序；

（6）删除程序；

（7）保护程序。

具体操作参看图4-26。



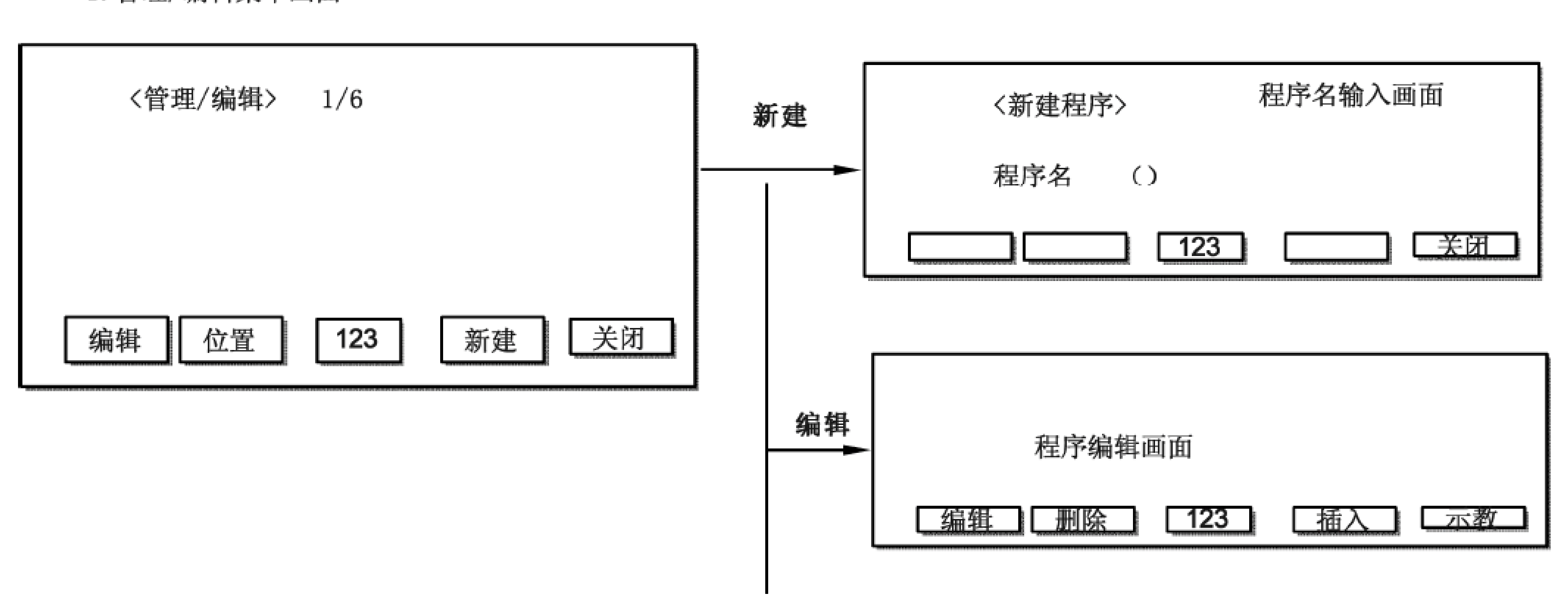


图4-26 主菜单

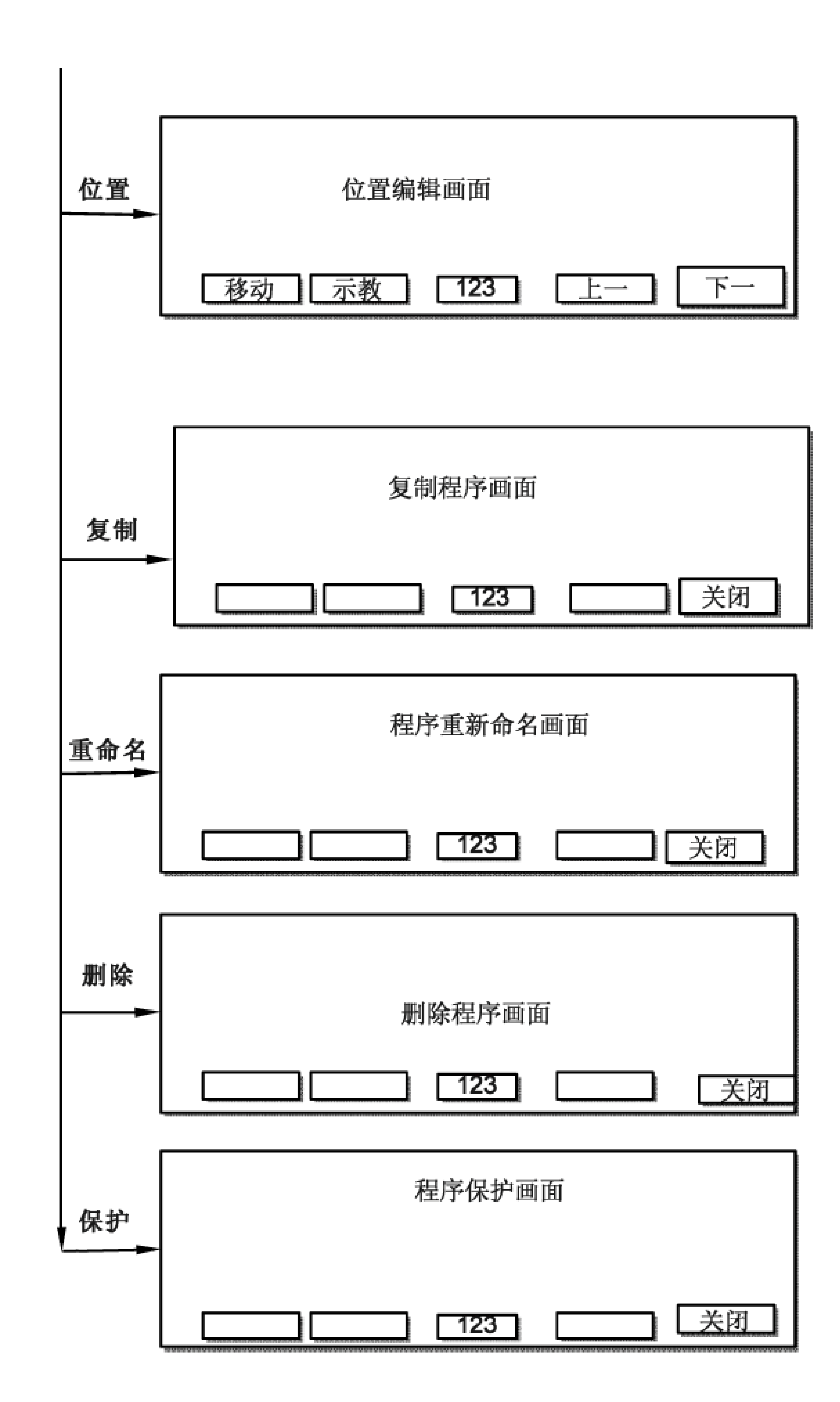


图 4-26 （续）

4.8.2 运行

本功能用于运行一个选定的程序，如图 4-27 所示。

1. 确认操作

确认用于执行某一程序行的动作，确认该程序行是否正确，是否到达预定的位置。

确认操作界面有以下功能。

(1) 前进——按顺序选择程序行。

(2) 跳转——跳转到指定的程序行。

(3) 后退——反向运行。

2. 测试运行——对选择的程序进行测试

有“连续运行模式”和“循环运行模式”两种模式。

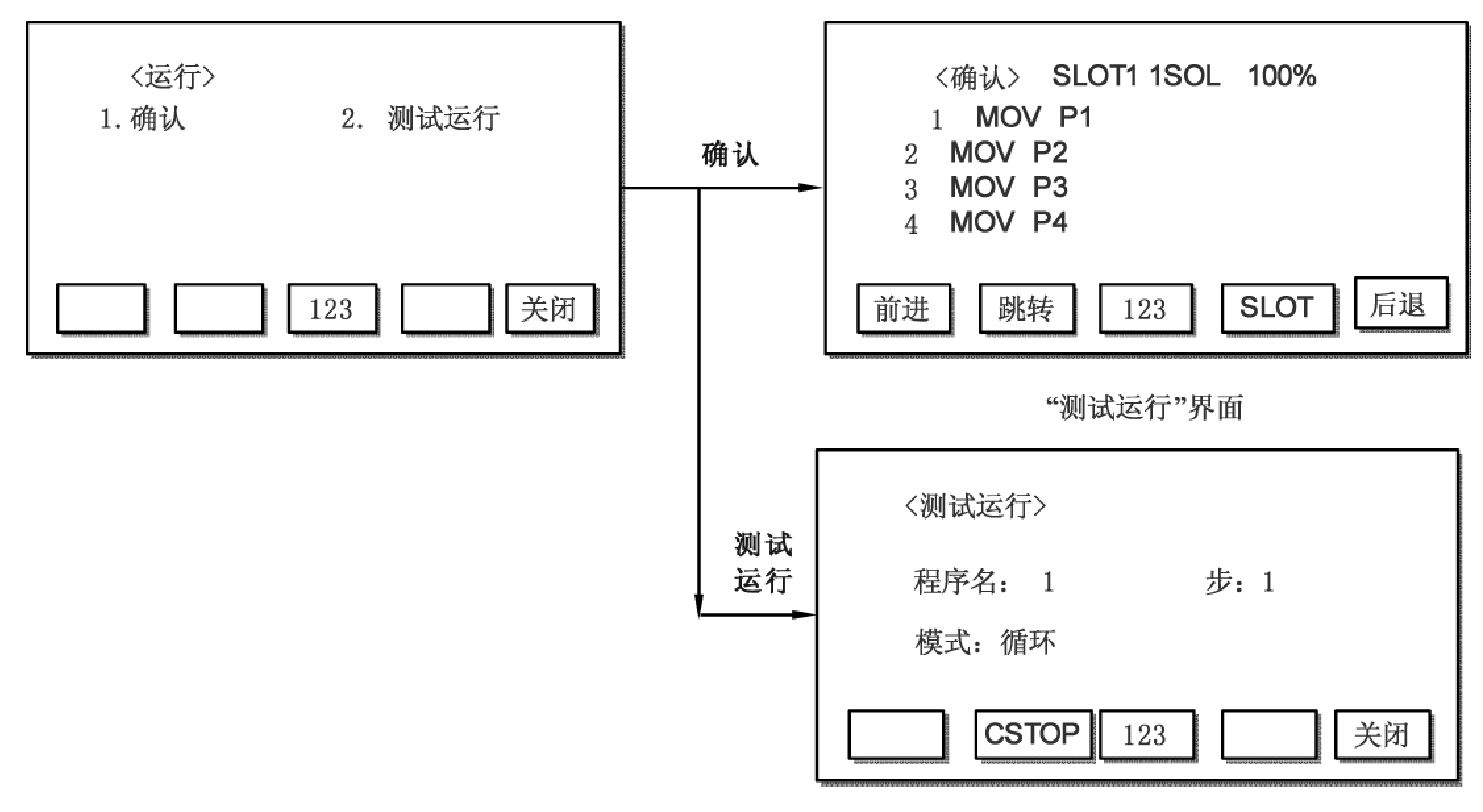


图 4-27 “运行”界面

4.8.3 参数的设置修改

本界面功能用于设置和修改参数，如图 4-28 所示。



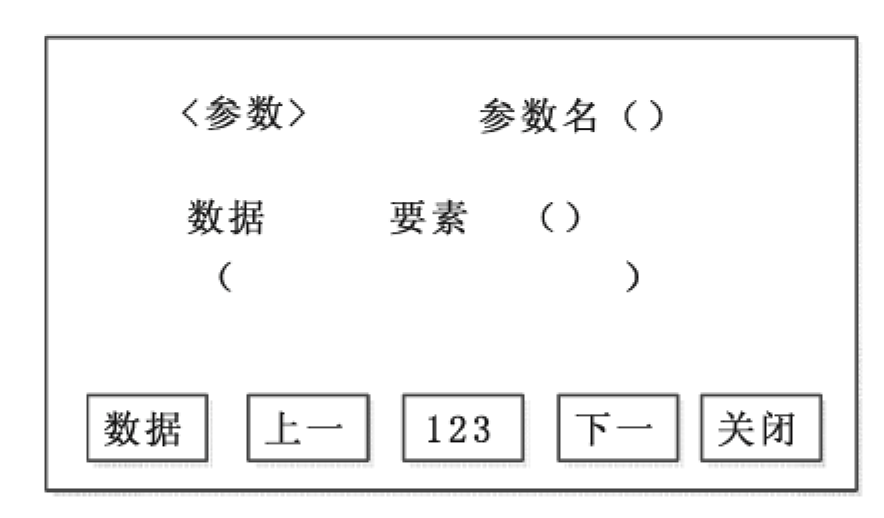


图 4-28 “参数”界面

4.8.4 原点设置/制动器设置

本界面用于设置原点和制动器，如图4-29所示。

(1) 原点设置：设置原点有6种方法(参见30.5.1节)。

(2) 制动器设置：制动器设置就是是否解除各伺服电机的抱闸功能。由于解除抱闸有危险，可能导致机器人某个轴坠落，所以必须特别小心。

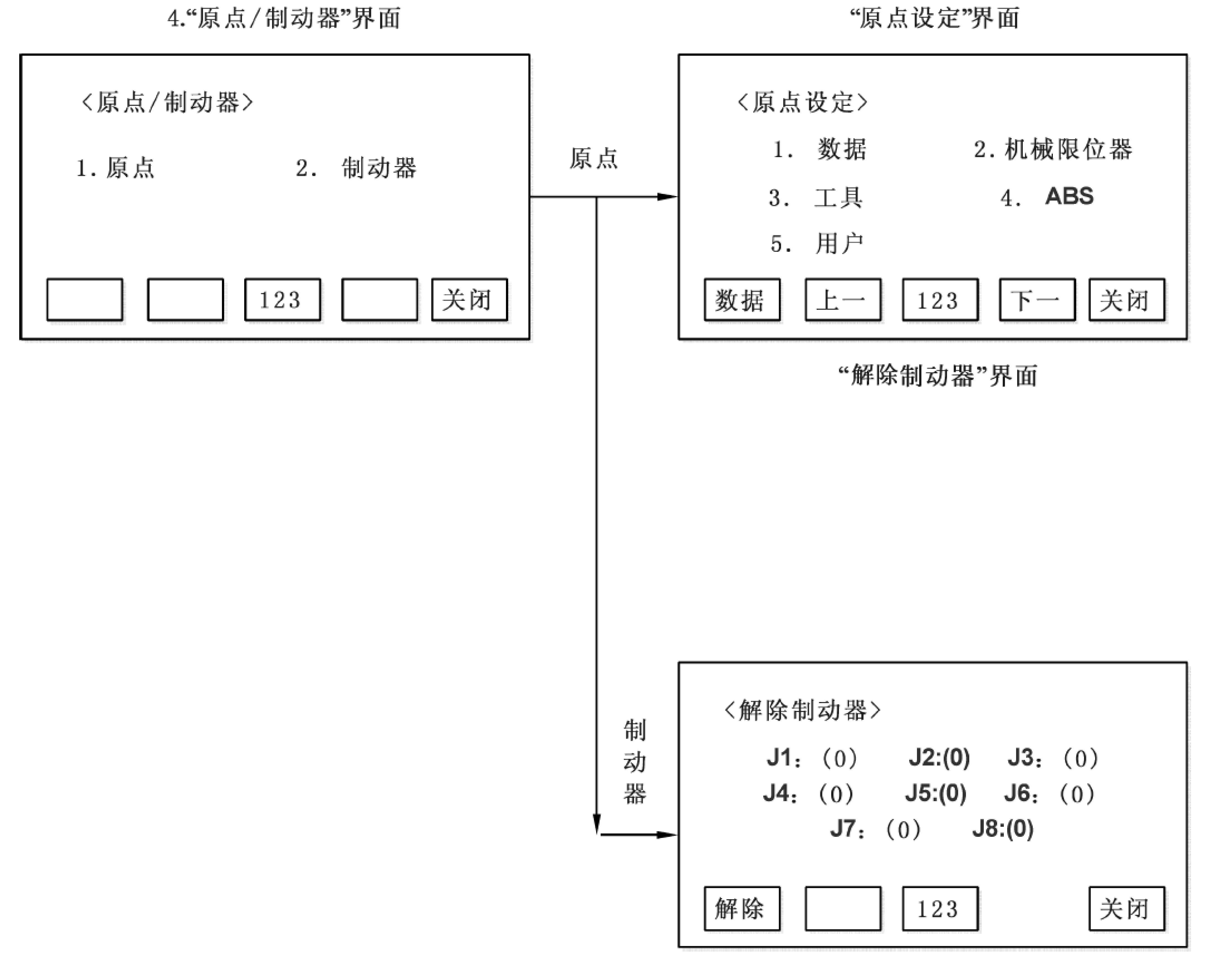


图4-29 原点设置功能界面

4.8.5 设定/初始化

初始化

初始化指对程序、参数、电池使用时间的初始化。

(1) 程序初始化——删除所有程序。

(2) 参数初始化——参数回到出厂值。

(3) 电池——清除电池使用时间。

操作方法参见图4-30。

5."设定/初始化"界面

〈设定/初始化〉

1. 初始化　　2. 运行

3. 时间设定　　4. 版本

123　关闭

初始化

1."初始化"界面

初始化

1. 程序　　2. 参数

3. 电源

数据　上一　123　下一　关闭

运行

2."运行时间"界面

〈运行时间〉

电源ON累计时间　45hr

电池剩余时间　14447hr

J7：（0）　J8:(0)

123　关闭

时间设定

3."时间设定"界面

〈时间设定〉

日期　17-06-04

时间　20:45:30

123　关闭

版本

4."版本"界面

〈版本〉

控制器：ver

示教单元：　ver

123　关闭

图 4-30　初始化功能界面

4.9　需要思考的问题

（1）如何执行调试程序的单步运行、跳转运行、后退运行？

（2）如何进行初始化设置？

（3）如何进行原点设置？

第 5 章　第 5 日——机器人系统的输入输出信号

【学习目的】

在一条实际的以机器人为中心构成的生产线中,会有大量的检测信号,比如“工件运行到位”“检测完毕”“压力不足”等信号,这些信号要进入到机器人控制器中供机器人使用。怎样接入这些信号在 2.7 节中已经做了说明,本章的学习内容是如何定义机器人内置的专用信号,如何使用外部 I/O 信号。

5.1　专用 I/O 信号

在机器人控制器中,为了方便用外部信号对机器人进行控制,预先设计配置了很多的“输入输出”的功能。例如,输入信号有“自动启动”“停止”,输出信号有“转矩到达”“发生碰撞”等。在机器人使用前,需要将这些功能分配设置到“外部 I/O 卡”的各输入输出点(通过参数设置),如图 5-1 所示。

输入信号(I)			输出信号(U)		
可自动运行	AUTOENA		可自动运行	AUTOENA	
启动	START	3	运行中	START	0
停止	STOP	0	待机中	STOP	
停止(STOP2)	STOP2		待机中2	STOP2	
			停止输入中	STOPSTS	
程序复位	SLOTINIT		可以选择程序	SLOTINIT	
报错复位	ERRRESET	2	报警发生中	ERRRESET	2
周期停止	CYCLE		周期停止中	CYCLE	
伺服OFF	SRVOFF	1	伺服ON不可	SRVOFF	
伺服ON	SRVON	4	伺服ON中	SRVON	1
操作权	IOENA	5	操作权	IOENA	3

图 5-1　通过参数给输入输出端子分配功能

在未进行设置前,“外部 I/O 卡”的各输入输出点是没有功能的(空白的)。

由于外部输入输出信号是每一套机器人系统都必须使用的信号,是每一套机器人系统的基本设置,本章将详细介绍输入输出信号的功能及设置。

5.2　输入输出信号的分类

机器人使用的输入输出信号分类如下。

(1) 专用输入输出信号——这是机器人系统内置的输入输出功能(信号)。这类信号功

能已经由系统内部规定但是具体分配到某个输入输出端子还需要由参数设置。这是使用最多的信号。

(2) 通用输入输出信号——这类信号例如“工件到位”“定位完成”由设计者自行定义，只与工程要求相关。

(3) 抓手信号——与抓手相关的输入输出信号。

5.3 专用输入输出信号详解

5.3.1 专用输入输出信号一览表

在机器人系统中，专用输入输出(某一功能)的“名称(英文)”是一样的，即同一“名称(英文)”可能表示输入也可能表示输出，初学者开始阅读指令手册时会感到困惑。本章将输入输出信号单独列出，便于读者阅读和使用。

1. 专用输入信号一览表

表 5-1 是专用输入信号功能一览表，这一部分信号是在实际工程中经常使用的。

表 5-1 专用输入信号一览表

序号	输入信号	功能简述	英文简称	出厂设定端子号
1	操作权	使外部信号操作权有效/无效	IOENA	5 *
2	启动	程序启动	START	3 *
3	停止	程序停止	STOP	0 (固定不变)
4	停止 2	程序停止。功能与 STOP 相同。但输入端子号可以任意设置	STOP2	*
5	程序复位	中断正执行的程序，回到程序起始行。对应多任务状态，使全部任务区程序复位	SLOTINIT	*
6	报警复位	解除报警状态	ERRRESET	*
7	伺服 ON	机器人伺服电源＝ON。多机器人时，全部机器人伺服电源＝ON	SRVON	*
8	伺服 OFF	机器人伺服电源＝OFF。多机器人时，全部机器人伺服电源＝OFF	SRVOFF	*
9	自动模式使能	使自动程序生效。禁止在非自动模式下做自动运行	AUTOENA	*
10	停止循环运行	停止循环运行的程序	CYCLE	*
11	机械锁定	使机器人进入机械锁定状态	MELOCK	*
12	回待避点	回到预设置的“待避点”	SAFEPOS	*
13	通用输出信号复位	指令全部“通用输出信号复位”	OUTRESET	*
14	第 N 任务区内程序启动	指令“第 N 任务区内程序启动”，N＝1～32	SnSTART	*
15	第 N 任务区内程序停止	指令“第 N 任务区内程序停止”，N＝1～32	SnSTOP	*
16	第 N 台机器人伺服电源 OFF	指令“第 N 台机器人伺服电源 OFF”，N＝1～3	SnSRVOFF	*

续表

序号	输入信号	功能简述	英文简称	出厂设定端子号
17	第N台机器人伺服电源ON	指令“第N台机器人伺服电源ON”，N=1～3	SnSRVON	*
18	第N台机器人机械锁定	指令“第N台机器人机械锁定”，N=1～3	SnMELOCK	*
19	选定程序生效	本信号用于使“选定的程序号”生效	PRGSEL	*
20	“选定速度比例”生效	本信号用于使“选定的速度比例”生效	OVRDSEL	*
21	数据输入	指定在选择“程序号”和“速度比例”等数据量时使用的输入信号“起始号”和“结束号”	IODATA	*
22	程序号输出请求	指令输出当前执行的“程序号”	PRGOUT	*
23	程序行号输出请求	指令输出当前执行的“程序行号”	LINEOUT	*
24	速度比例输出请求	指令输出当前“速度比例”	OVRDOUT	*
25	“报警号”输出请求	指令输出当前“报警号”	ERROUT	*
26	JOG使能信号	使JOG功能生效(通过外部端子使用JOG功能)	JOGENA	*
27	用数据设置JOG运行模式	设置在选择“JOG模式”时使用的端子“起始号”和“结束号”。0/1/2/3/4=关节/直交/圆筒/三轴直交/TOOL	JOGM	*
28	JOG+	指定各轴的JOG+信号	JOG+	*
29	JOG−	指定各轴的JOG−信号	JOG−	*
30	工件坐标系编号	通过数据“起始位”与“结束位”设置“工件坐标系编号”	JOGWKND	*
31	JOG报警暂时无效	本信号=ON，JOG报警暂时无效	JOGNER	*
32	是否允许外部信号控制抓手	本信号=ON/OFF，允许/不允许外部信号控制抓手	HANDENA	*
33	控制抓手的输入信号范围	设置“控制抓手的输入信号范围”	HANDOUT	*
34	第N台机器人的抓手报警N=1～3	发出“第N台机器人抓手报警信号”	HNDERRn	*
35	第N台机器人的气压报警(N=1～5)	发出“第N台机器人的气压报警”信号	AIRERRn	*
36	第N台机器人预热运行模式有效	发出“第N台机器人预热运行模式有效”信号	MnWUPENA (N=1～3)	*
37	指定需要输出位置数据的“任务区”号	指定需要输出位置数据的“任务区”号	PSSLOT	*
38	位置数据类型	指定位置数据类型 I/O=关节型变量/直交型变量	PSTYPE	*
39	指定用一组数据表示“位置变量号”	指定用一组数据表示“位置变量号”	PSNUM	*
40	输出位置数据指令	指令输出当前“位置数据”	PSOUT	*
41	输出控制柜温度	指令输出控制柜实际温度	TMPOUT	*

*表示可以由用户自行设置输入端子号。

2. 专用输出信号一览表

在机器人系统中，对于同一功能，输入输出信号的“英文简称”是相同的。但是输入信号的功能是使得这一功能起作用，输出信号的功能是表示这一功能已经起作用。专用输出信号大多是表示机器人系统的工作状态。表 5-2 所示是专用输出信号一览表。

表 5-2 专用输出信号一览表

序号	输出信号	功 能	英文简称	出厂设置
1	控制器电源 ON	表示控制器电源 ON，可以正常工作	RCREADY	*
2	远程模式	表示操作面板选择自动模式，外部 I/O 信号操作有效	ATEXTMD	*
3	示教模式	表示当前工作模式为“示教模式”	TEACHMD	*
4	自动模式	表示当前工作模式为“自动模式”	ATTOPMD	*
5	外部信号操作权有效	表示“外部信号操作权有效”	IOENA	3
6	程序已启动	表示机器人进入“程序已启动”状态	START	*
7	程序停止	表示机器人进入“程序暂停”状态	STOP	*
8	程序停止	表示当前为“程序暂停”状态	STOP2	*
9	STOP 信号输入	表示正在输入 STOP 信号	STOPSTS	*
10	任务区中的程序可选择状态	表示“任务区处于程序可选择状态”	SLOTINIT	*
11	报警发生中	表示系统处于“报警发生”状态	ERRRESET	*
12	伺服 ON	表示系统当前处于“伺服 ON”状态	SRVON	1
13	伺服 OFF	表示系统当前处于“伺服 OFF”状态	SRVOFF	*
14	可自动运行	表示系统当前处于“可自动运行”状态	AUTOENA	*
15	循环停止信号	表示“循环停止信号”正输入中	CYCLE	*
16	机械锁定状态	表示机器人处于“机械锁定状态”	MELOCK	*
17	回归待避点状态	表示机器人处于“回归待避点状态”	SAFEPOS	*
18	电池电压过低	表示机器人“电池电压过低”	BATERR	*
19	严重级报警	表示机器人出现“严重级故障报警”	HLVLERR	*
20	轻量级故障报警	表示机器人出现“轻量级故障报警”	LLVLERR	*
21	警告型故障	表示机器人出现“警告型故障”	CLVLERR	*
22	机器人急停	表示机器人处于“急停状态”	EMGERR	*
23	第 N 任务区程序在运行中	表示“第 N 任务区程序在运行中”	SnSTART	*
24	第 N 任务区程序在暂停中	表示“第 N 任务区程序在暂停中”	SnSTOP	*
25	第 N 台机器人伺服 OFF	表示“第 N 台机器人伺服 OFF”	SnSRVOFF	*
26	第 N 台机器人伺服 ON	表示“第 N 台机器人伺服 ON”	SnSRVON	*
27	第 N 台机器人机械锁定	表示“第 N 台机器人处于机械锁定”状态	SnMELOCK	*
28	数据输出区域	对数据输出，指定输出信号的“起始位”和“结束位”	IODATA	*
29	“程序号”数据输出中	表示当前正在输出“程序号”	PRGOUT	*
30	“程序行号”数据输出中	表示当前正在输出“程序行号”	LINEOUT	*
31	“速度比例”数据输出中	表示当前正在输出“速度比例”	OVRDOUT	*
32	“报警号”输出中	表示当前正在输出“报警号”	ERROUT	*
33	JOG 有效状态	表示当前处于“JOG 有效状态”	JOGENA	*
34	JOG 模式	表示当前的“JOG 模式”	JOGM	*
35	当前工件坐标系编号	显示“当前工件坐标系编号”	JOGWKND	*
36	抓手工作状态	输出抓手工作状态（输出信号部分）	HNDCNTLn	*
37	抓手工作状态	输出抓手工作状态（输入信号部分）	HNDSTSn	*

续表

序号	输出信号	功　　能	英文简称	出厂设置
38	外部信号对抓手控制的有效/无效状态	表示“外部信号对抓手控制的有效/无效状态”	HANDENA	*
39	第N台机器人抓手报警	表示“第N台机器人抓手报警”	HNDERRn	*
40	第N台机器人气压报警	表示“第N台机器人气压报警”	AIRERRn	*
41	用户定义区编号	用输出端子“起始位”“结束位”表示“用户定义区编号”	USRAREA	*
42	易损件维修时间	表示易损件到达“维修时间”	MnPTEXC	*
43	机器人处于“预热工作模式”	表示“机器人处于预热工作模式”	MnWUPENA	*
44	输出位置数据的任务区编号	用输出端子“起始位”“结束位”表示“输出位置数据的任务区编号”	PSSLOT	*
45	输出的“位置数据类型”	表示输出的“位置数据类型”是关节型还是直交型	PSTYPE	*
46	输出的“位置数据编号”	用输出端子“起始位”“结束位”表示“输出位置数据的编号”	PSNUM	*
47	“位置数据”的输出状态	表示当前是否处于“位置数据的输出状态”	PSOUT	*
48	控制柜温度输出状态	表示当前处于“控制柜温度输出状态”	TMPOUT	*

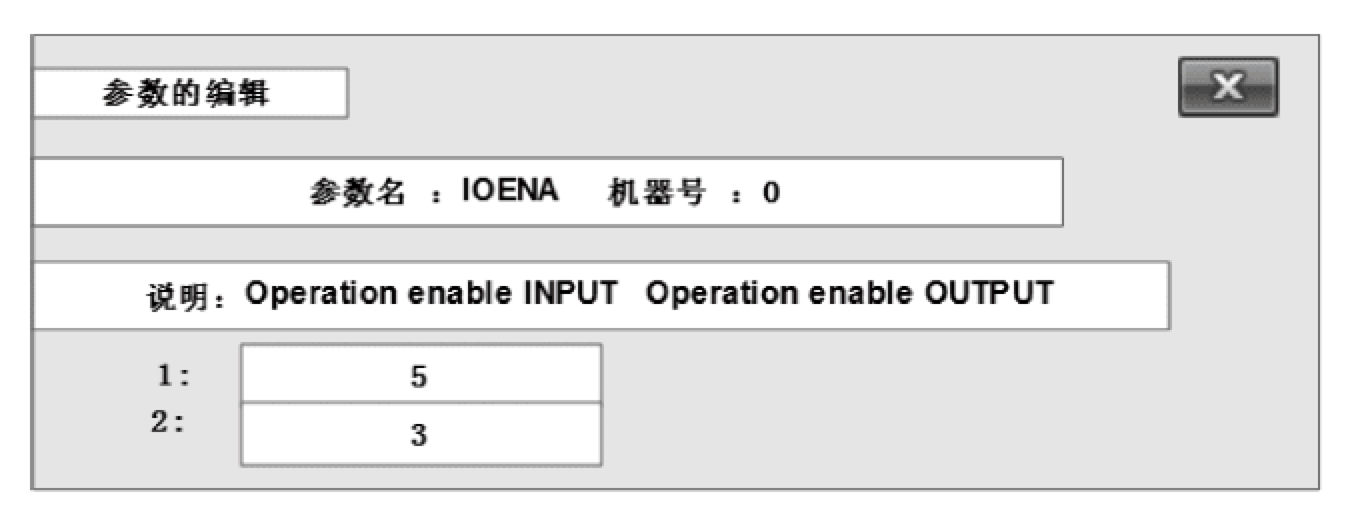
* 表示可以由用户自行设置输出端子号。

5.3.2　专用输入信号详解

本节解释“专用输入信号”以及这些信号对应的参数。出厂值是指出厂时预分配的输入端子编号。机器人系统本身已经内置了专用的“功能”，本节将对这些功能进行解释。使用时通过参数将这些功能赋予指定的“输入端子”，有些功能特别重要，所以出厂时已经预先设定了“输入端子编号”，即该输入端被指定了功能，不得更改(例如STOP功能)。如果出厂值=“－1”则表示可以任意设置“输入端子编号”。设置参数是通过软件RT ToolBox或示教单元进行的，所以本节使用了软件RT ToolBox的参数设置界面，这样更有助于读者对“专用功能”的理解，如表5-3所示。

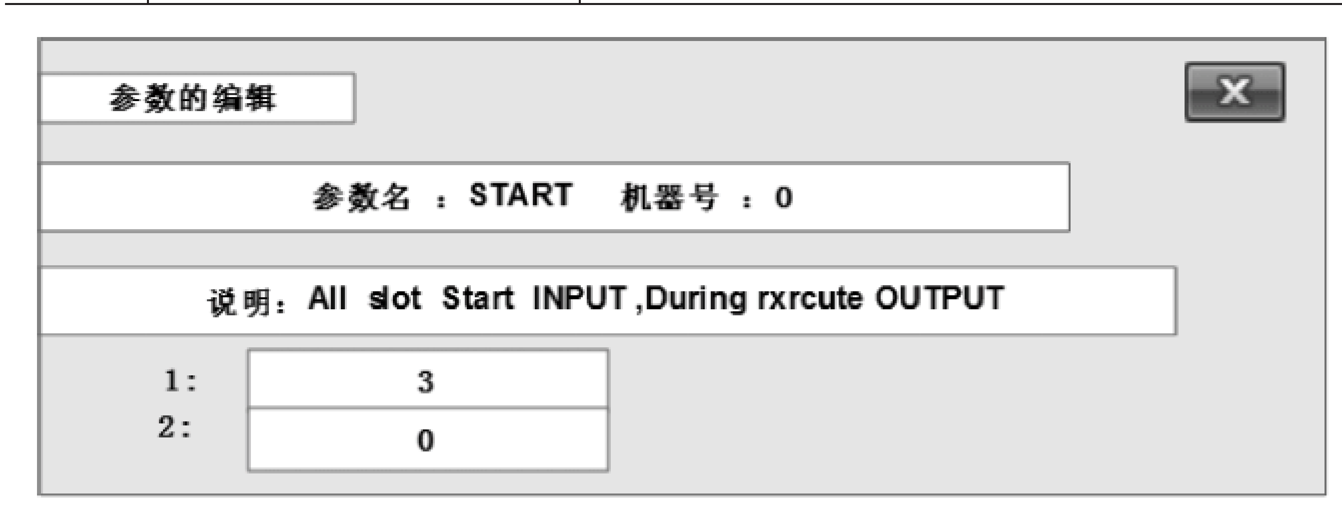
表5-3　输入信号

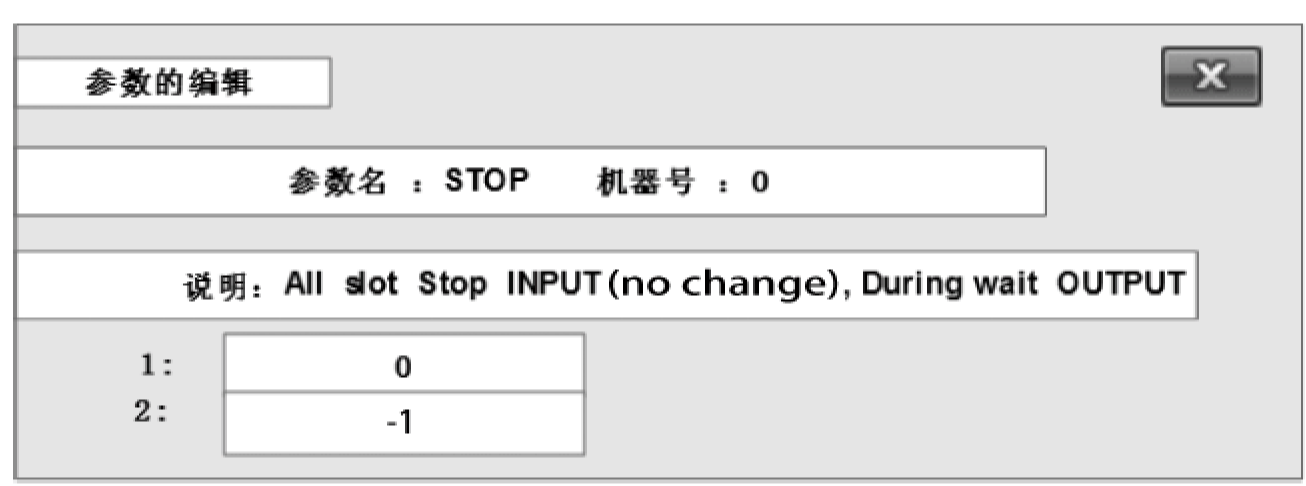
序号	名称	功　　能	对应参数	出厂值(端子号)
1	操作权	使外部信号操作权有效/无效	IOENA	5



图中，设置对应本功能的输入端子号=5，输入端子5=ON/OFF，对应“外部信号操作权”有效/无效。输入端子5=ON，从I/O卡输入的信号生效。输入端子5=OFF，从I/O卡输入的信号无效

续表

<table>
<tr><th>序号</th><th>名称</th><th>功　　能</th><th>对应参数</th><th>出厂值(端子号)</th></tr>
<tr><td>2</td><td>启动</td><td>程序启动</td><td>START</td><td>3</td></tr>
<tr><td colspan="5">

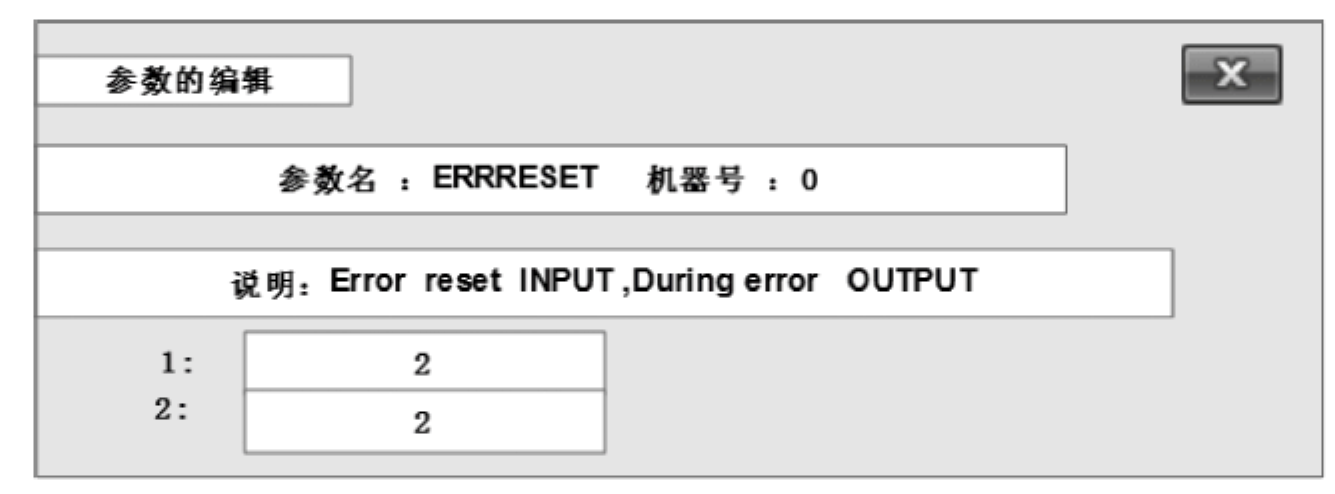
图中，设置对应本功能的输入端子号＝3，如输入端子 3＝ON，则所有任务区内程序启动</td></tr>
<tr><td>3</td><td>停止</td><td>程序停止</td><td>STOP</td><td>0(固定不变)</td></tr>
<tr><td colspan="5">

图中，设置对应本功能的输入端子号＝0，如输入端子 0＝ON，则所有任务区内“程序停止”。STOP 功能对应的输入端子号固定设置＝0</td></tr>
<tr><td>4</td><td>停止 2</td><td>程序停止。功能与 STOP 相同。但输入端子号可以任意设置</td><td>STOP2</td><td></td></tr>
<tr><td colspan="5">

图中，设置对应本功能的输入端子号＝8，如输入端子 8＝ON，则所有任务区内“程序停止”。STOP2 功能对应的输入端子号可以由用户设置</td></tr>
<tr><td>5</td><td>程序复位</td><td>中断正在执行的程序，回到程序起始行。对应多任务状态，使全部任务区程序复位。当对应启动条件为 ALWAYS 和 ERROR 时，则不能够执行复位</td><td>SLOTINIT</td><td></td></tr>
<tr><td colspan="5">参数的编辑
参数名 ：SLOTINIT 机器号 ：0
说明：Program reset INPUT ,Program select enable OUTPUT
1：6
2：-1
图中，设置对应本功能的输入端子号＝6，如输入端子 6＝ON，则所有任务区内“程序复位”</td></tr>
</table>

续表

序号	名称	功　　能	对应参数	出厂值(端子号)
6	报警复位	解除报警状态	ERRRESET	2



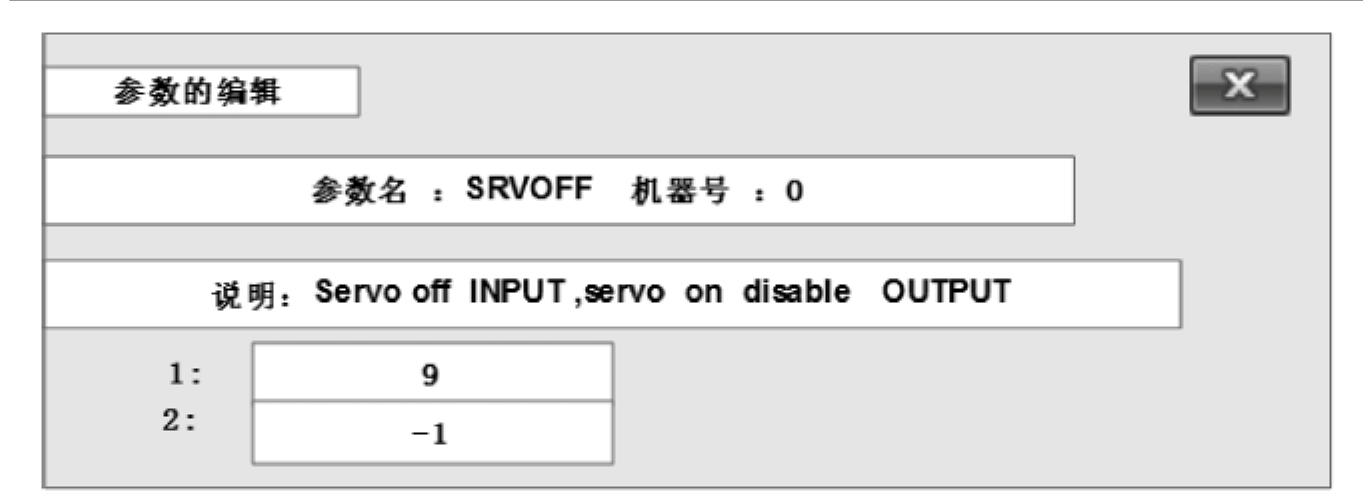
图中，设置对应本功能的输入端子号＝2，如输入端子 2＝ON，则解除报警状态

7	伺服 ON	机器人伺服电源＝ON 多机器人时，全部机器人伺服电源＝ON	SRVON	4

参数的编辑
参数名 ：SRVON　机器号 ：0
说明：Servo on INPUT,Duringservo on OUTPUT
1：4
2：1

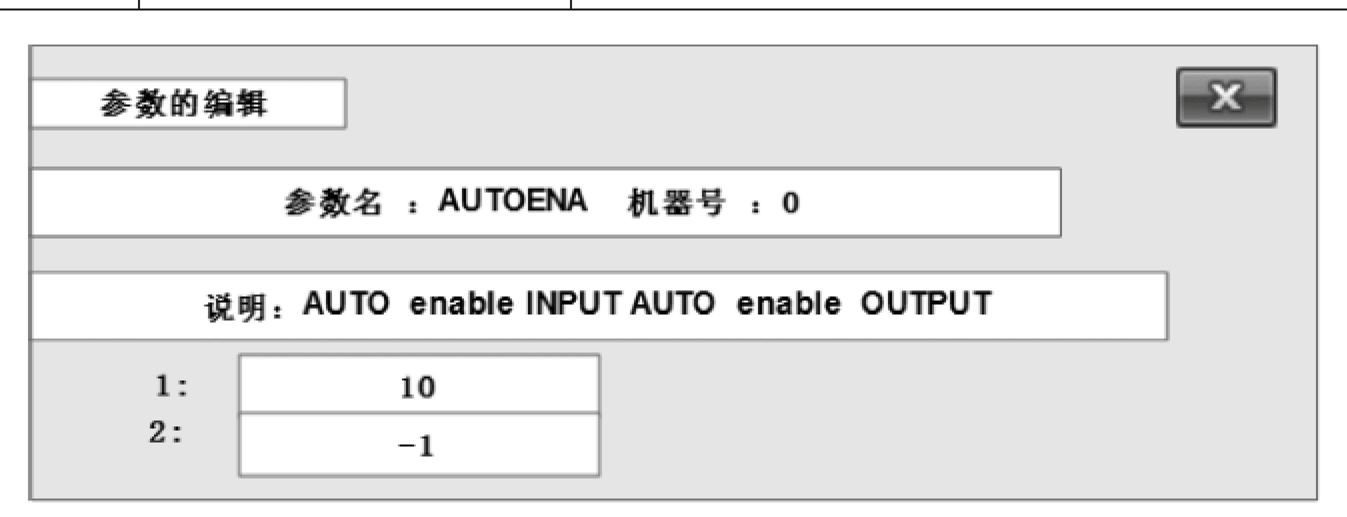
图中，设置对应本功能的输入端子号＝4，如输入端子 4＝ON，则机器人伺服电源＝ON

8	伺服 OFF	机器人伺服电源＝OFF 多机器人时，全部机器人伺服电源＝OFF	SRVOFF	



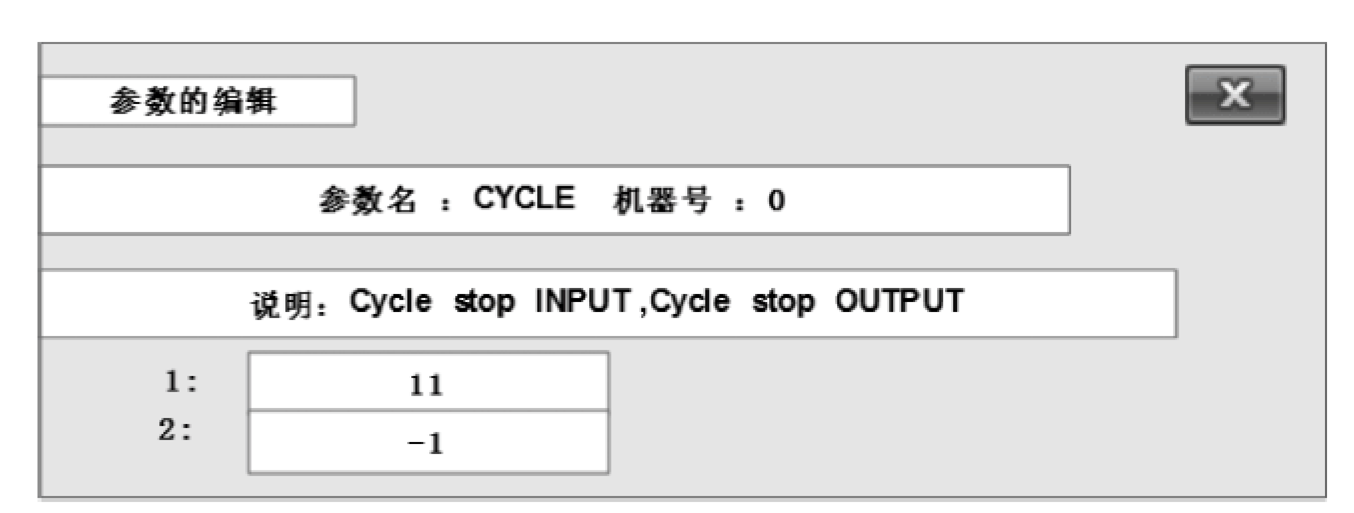
图中，设置对应本功能的输入端子号＝9，如输入端子 9＝ON，则机器人伺服电源＝OFF

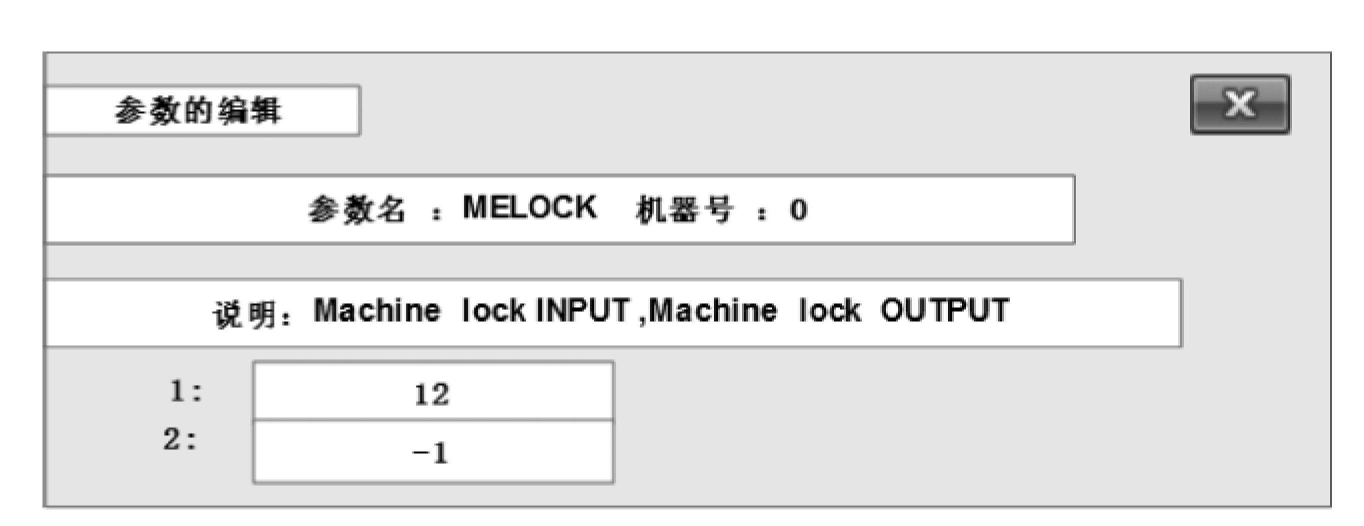
9	自动模式使能	使自动程序生效。禁止在非自动模式下做自动运行	AUTOENA	

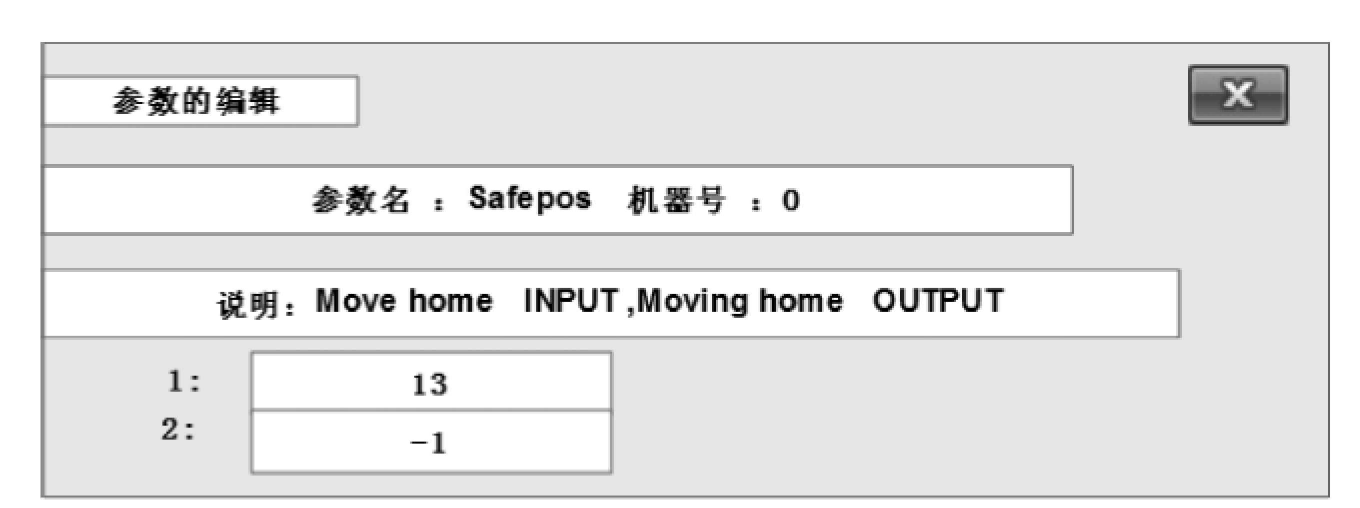


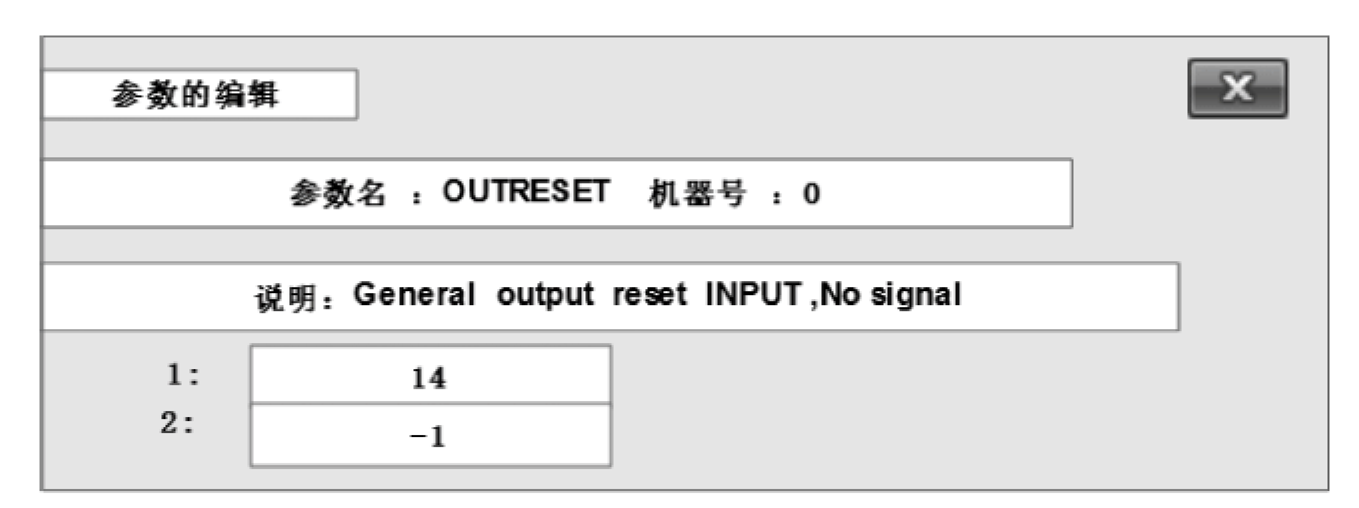
图中，设置对应本功能的输入端子号＝10，如输入端子 10＝ON，则机器人进入自动使能模式

续表

<table>
<tr><th>序号</th><th>名称</th><th>功　　能</th><th>对应参数</th><th>出厂值(端子号)</th></tr>
<tr><td>10</td><td>停止循环运行</td><td>停止循环运行的程序</td><td>CYCLE</td><td></td></tr>
<tr><td colspan="5">


图中,设置对应本功能的输入端子号=11,如输入端子 11=ON,则停止循环运行的程序</td></tr>
<tr><td>11</td><td>机械锁定</td><td>使机器人进入机械锁定状态。
机械锁定状态——程序运行,机械不动</td><td>MELOCK</td><td></td></tr>
<tr><td colspan="5">


图中,设置对应本功能的输入端子号=12,如输入端子 12=ON,则机械锁定功能生效</td></tr>
<tr><td>12</td><td>回待避点</td><td>回到预设置的“待避点”</td><td>SAFEPOS</td><td></td></tr>
<tr><td colspan="5">


图中,设置对应本功能的输入端子号=13,如输入端子 13=ON,则执行回待避点动作</td></tr>
<tr><td>13</td><td>通用输出信号复位</td><td>指令全部“通用输出信号复位”</td><td>OUTRESET</td><td></td></tr>
<tr><td colspan="5">


图中,设置对应本功能的输入端子号=14,如输入端子 14=ON,则执行“通用输出信号复位”动作</td></tr>
</table>

续表

序号	名称	功　　能	对应参数	出厂值(端子号)
14	第N任务区内程序启动	指令“第N任务区内程序启动”,N=1～32	SnSTART	

参数的编辑
参数名 ：S2START　机器号 ：0
说明：Slot2 Start INPUT , Slot2 During execute OUTPUT
1：15
2：-1

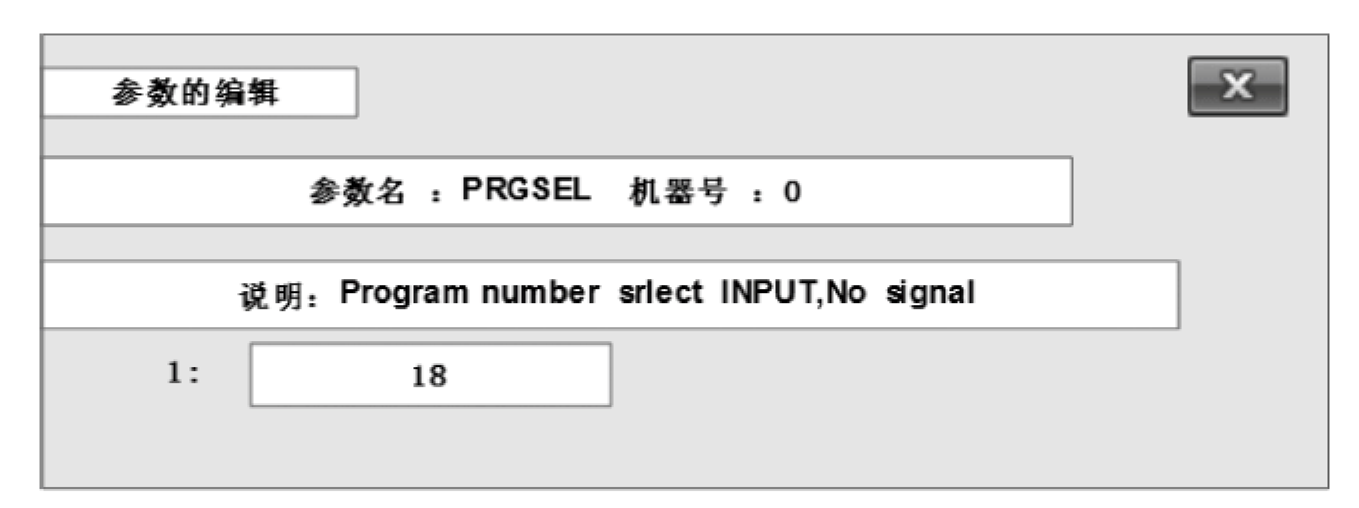
图中,设置对应本功能的输入端子号=15,如输入端子15=ON,则执行“第2任务区内程序启动”

15	第N任务区内程序停止	指令“第N任务区内程序停止”,N=1～32	SnSTOP	

参数的编辑
参数名 ：S2STOP　机器号 ：0
说明：Slot2 Stop INPUT, slot2 During wait OUTPUT
1：16
2：-1

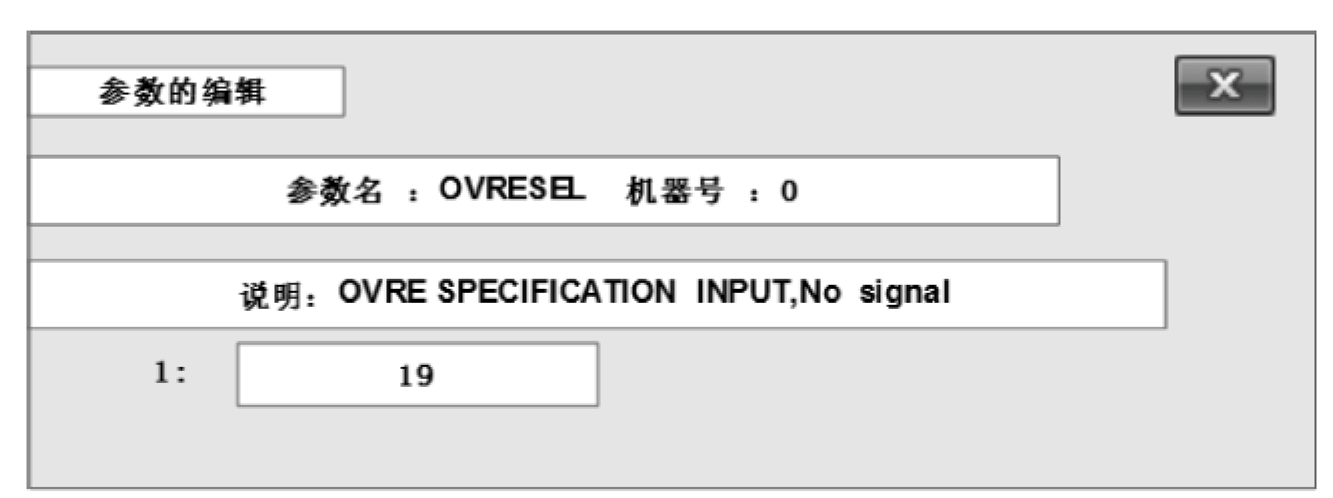
图中,设置对应本功能的输入端子号=16,如输入端子16=ON,则执行“第2任务区内程序停止”

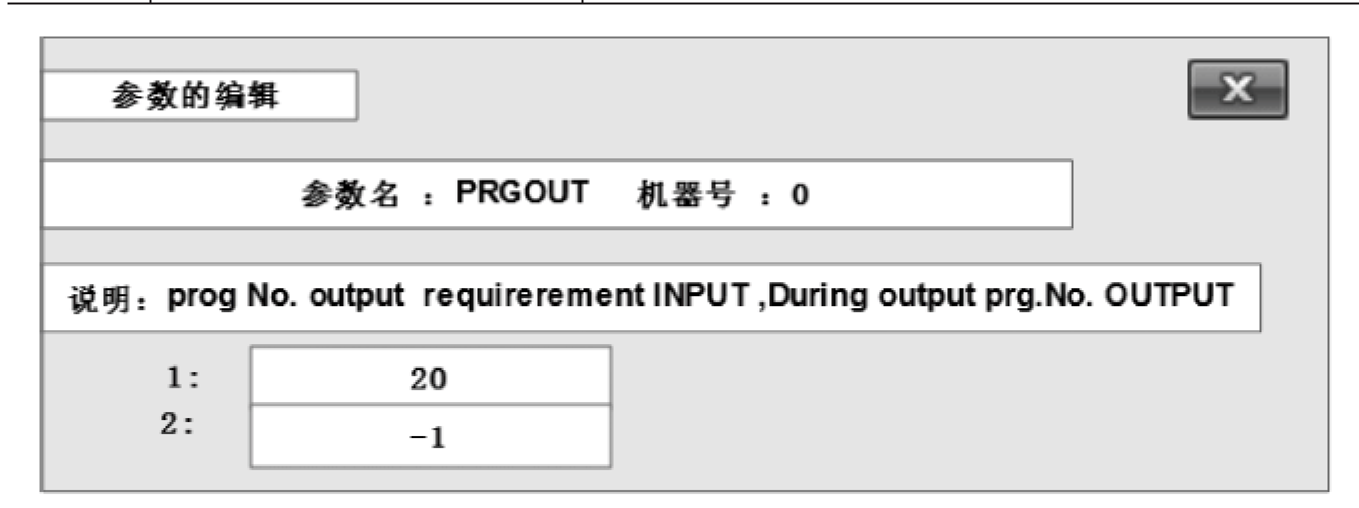
16	第N台机器人伺服电源OFF	指令“第N台机器人伺服电源OFF”,N=1～3	SnSRVOFF	
17	第N台机器人伺服电源ON	指令“第N台机器人伺服电源ON”,N=1～3	SnSRVON	
18	第N台机器人机械锁定	指令“第N台机器人机械锁定”,N=1～3	SnMELOCK	
19	选定程序生效	本信号用于使“选定的程序号”生效	PRGSEL	

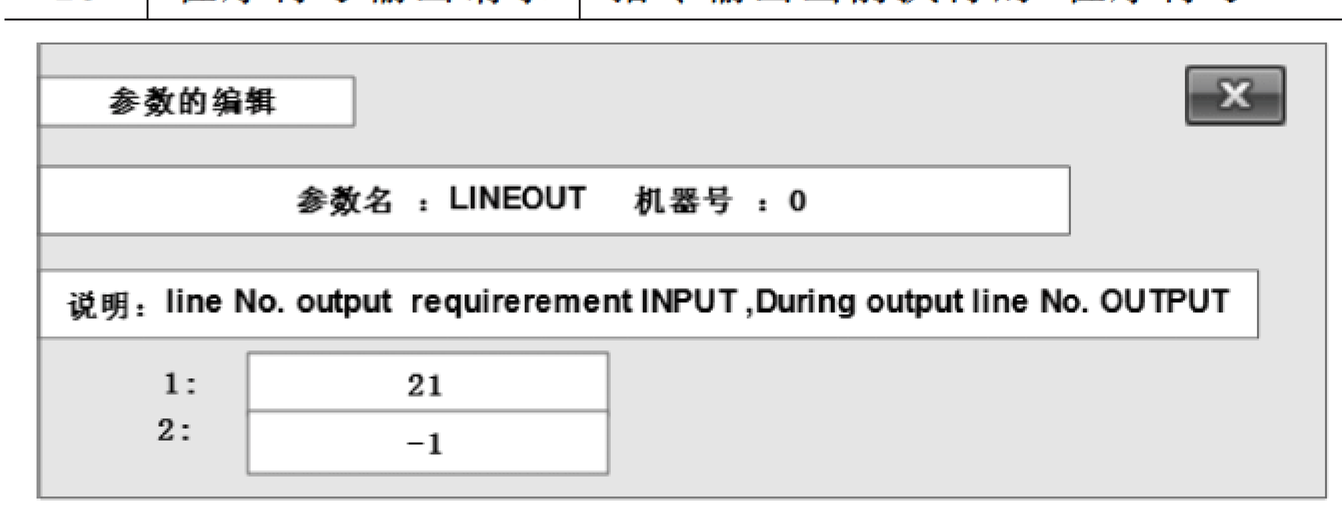


图中,设置对应本功能的输入端子号=18,如输入端子18=ON,则“选定的程序号”生效

续表

<table>
<tr><th>序号</th><th>名称</th><th>功　　能</th><th>对应参数</th><th>出厂值(端子号)</th></tr>
<tr><td>20</td><td>“选定速度比例”生效</td><td>本信号用于使“选定的速度比例”生效</td><td>OVRDSEL</td><td></td></tr>
<tr><td colspan="5">

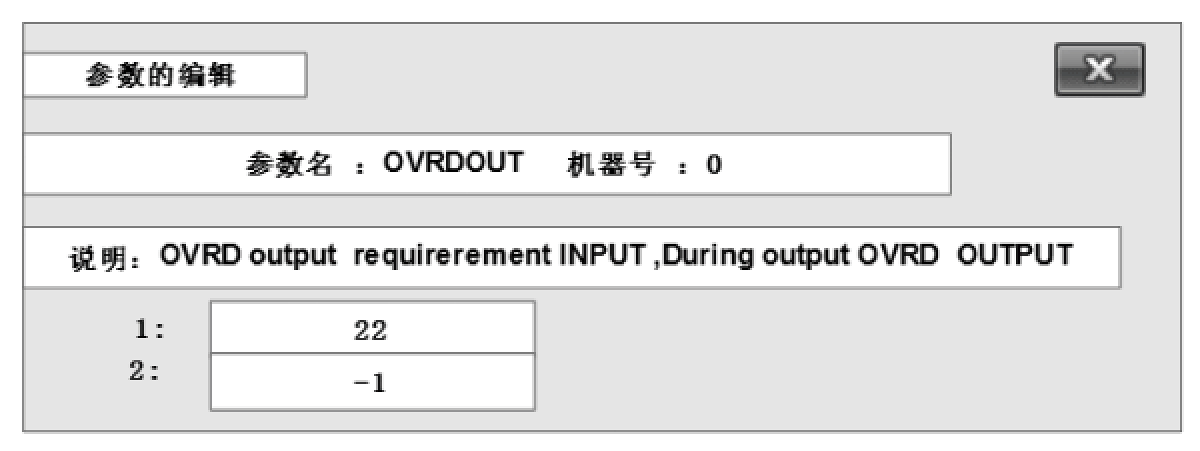
图中，设置对应本功能的输入端子号＝19，如输入端子 19＝ON，则“选定速度比例”生效</td></tr>
<tr><td>21</td><td>数据输入</td><td>指定在选择“程序号”和“速度比例”等数据量时使用的输入信号“起始号”和“结束号”</td><td>IODATA</td><td></td></tr>
<tr><td colspan="5">参数的编辑
参数名 ：IODATA　机器号 ：0
说明：Value input signal (start,end)INPUT ,Value output signal (start,end)OUTPUT
1：12
2：15
3　12
4　15
图中，设置对应本功能的输入端子号＝12～15，输入端子号＝12～15 组成的(二进制)数据可以为“程序号”“速度比例”等数据输入量</td></tr>
<tr><td>22</td><td>程序号输出请求</td><td>指令输出当前执行的“程序号”</td><td>PRGOUT</td><td></td></tr>
<tr><td colspan="5">

图中，设置对应本功能的输入端子号＝20，如输入端子 20＝ON，则指令输出当前执行的“程序号”</td></tr>
<tr><td>23</td><td>程序行号输出请求</td><td>指令输出当前执行的“程序行号”</td><td>LINEOUT</td><td></td></tr>
<tr><td colspan="5">

图中，设置对应本功能的输入端子号＝21，如输入端子 21＝ON，则指令输出当前执行的“程序行号”</td></tr>
</table>

续表

序号	名称	功　　能	对应参数	出厂值（端子号）
24	速度比例输出请求	指令输出当前“速度比例”	OVRDOUT	



图中，设置对应本功能的输入端子号＝22，如输入端子22＝ON，则指令输出当前执行的“速度比例”

序号	名称	功　　能	对应参数	出厂值（端子号）
25	“报警号”输出请求	指令输出当前“报警号”	ERROUT	

参数的编辑
参数名 ：ERROUT　机器号 ：0
说明：Err No. output requirerement INPUT ,During output Err. NO. OUTPUT
1：23
2：-1

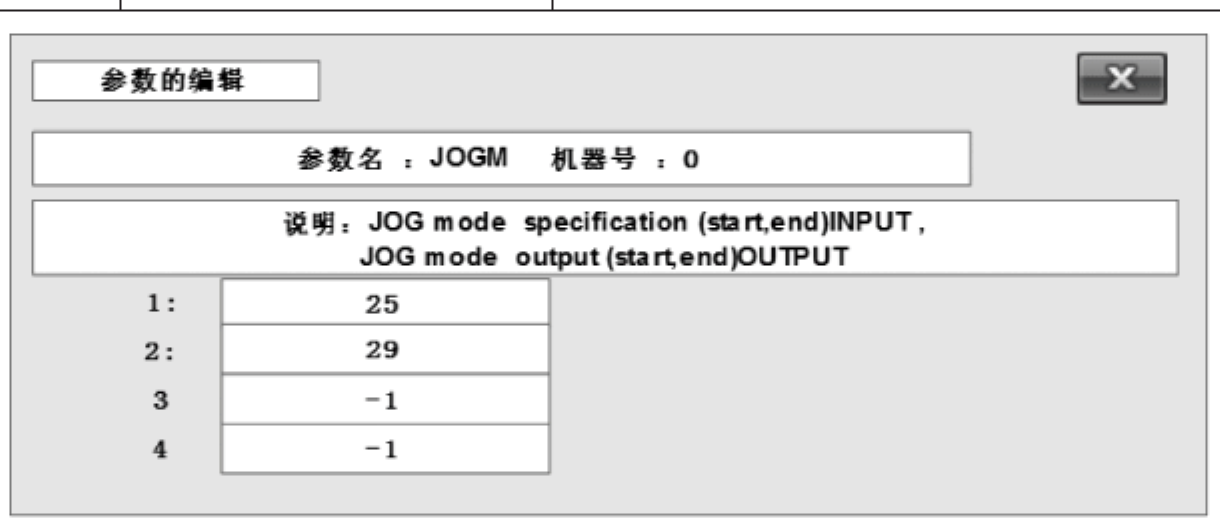
图中，设置对应本功能的输入端子号＝23，如输入端子23＝ON，则指令输出当前的“报警号”

序号	名称	功　　能	对应参数	出厂值（端子号）
26	JOG使能信号	使JOG功能生效（通过外部端子使用JOG功能）	JOGENA	

参数的编辑
参数名 ：JOGENA　机器号 ：0
说明：JOG command INPUT ,During JOG OUTPUT
1：24
2：-1

图中，设置对应本功能的输入端子号＝24，如输入端子24＝ON，则JOG功能生效（通过外部端子使用JOG功能）

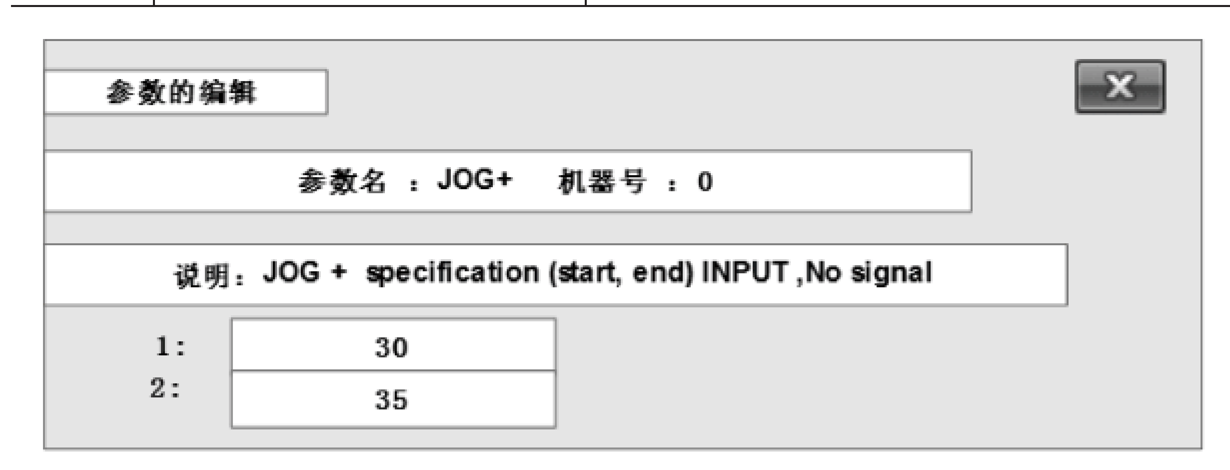
序号	名称	功　　能	对应参数	出厂值（端子号）
27	用数据设置JOG运行模式	设置在选择“JOG模式”时使用的端子“起始号”和“结束号” 0/1/2/3/4＝关节/直交/圆筒/三轴直交/TOOL	JOGM	



图中，设置对应本功能的输入端子号＝25～29，输入端子号＝25～29组成的数据为“JOG运行的工作模式”。0/1/2/3/4＝关节/直交/圆筒/三轴直交/TOOL。例如：
输入端子号＝25～29组成的数据＝1，则选择直交模式

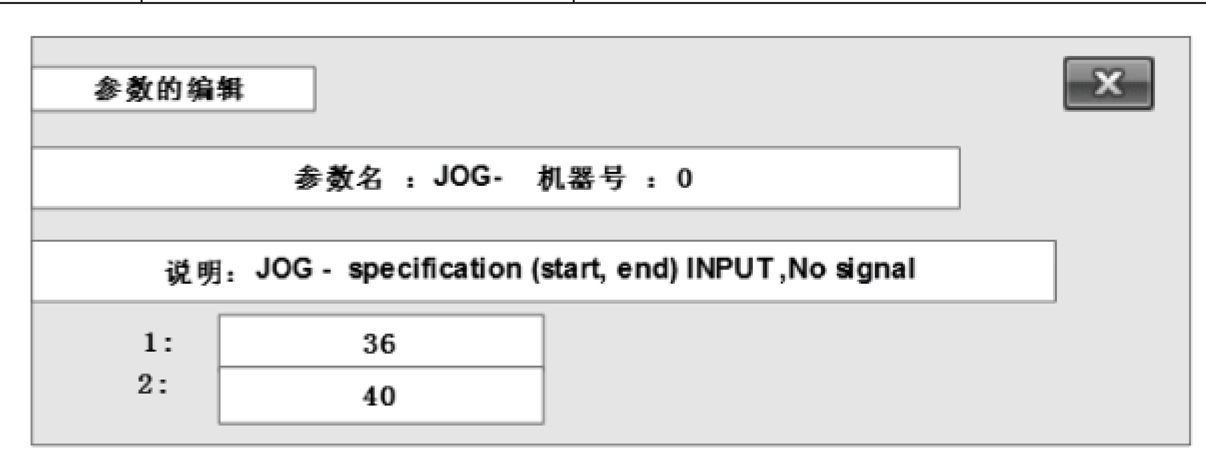
续表

序号	名称	功　　能	对应参数	出厂值(端子号)
28	JOG＋	指定各轴的 JOG＋信号	JOG＋	



图中，设置对应本功能的输入端子号＝30～35，即输入端子 30＝J1 轴 JOG＋，输入端子 31＝J2 轴 JOG＋，……输入端子 35＝J6 轴 JOG＋

序号	名称	功　　能	对应参数	出厂值(端子号)
29	JOG－	指定各轴的 JOG－信号	JOG－	



图中，设置对应本功能的输入端子号＝36～40，即输入端子 36＝J1 轴 JOG－，输入端子 37＝J2 轴 JOG－，……输入端子 40＝J6 轴 JOG－

序号	名称	功　　能	对应参数	出厂值(端子号)
30	工件坐标系编号	通过数据"起始位"与"结束位"设置"工件坐标系编号"	JOGWKND	
31	JOG 报警暂时无效	本信号＝ON，JOG 报警暂时无效	JOGNER	

参数的编辑
参数名 ：JOGNER　机器号 ：0
说明：Error disregard at JOG INPUT ,Durind Error disregard at JOG OUTPUT
1：　41
2：　-1

图中，设置对应本功能的输入端子号＝41，如输入端子 41＝ON，则 JOG 报警暂时无效

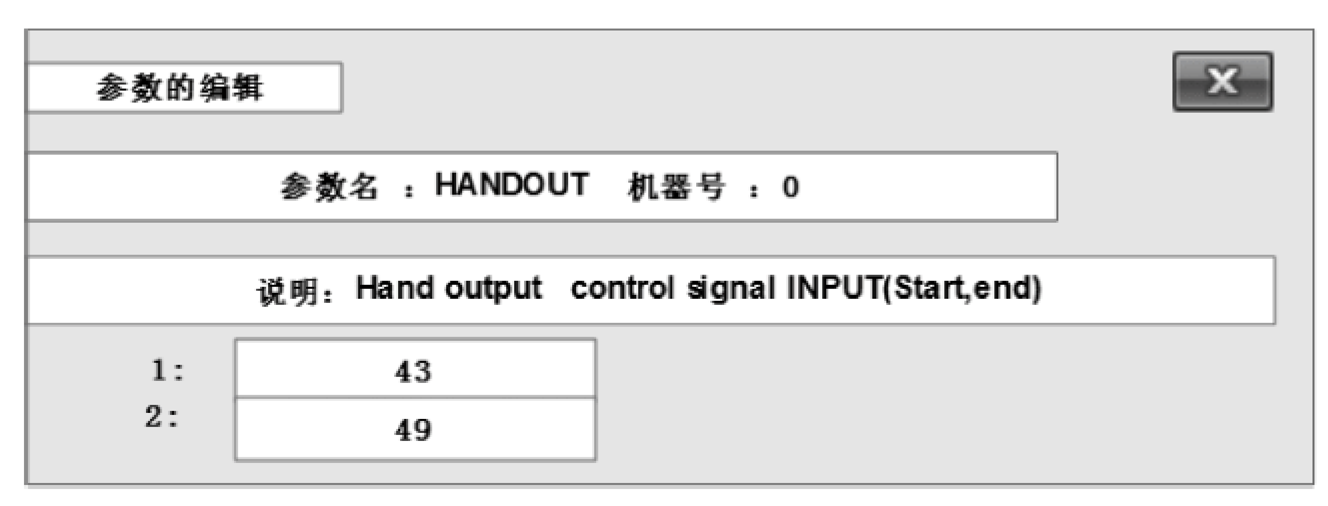
序号	名称	功　　能	对应参数	出厂值(端子号)
32	是否允许外部信号控制抓手	本信号＝ON/OFF，允许/不允许外部信号控制抓手	HANDENA	

参数的编辑
参数名 ：HANDENA　机器号 ：0
说明：Hand control enable INPUT ,Hand control enable OUTPUT
1：　41
2：　-1

图中，设置对应本功能的输入端子号＝42，如输入端子 42＝ON，则允许外部信号控制抓手。如输入端子 42＝OFF，则不允许外部信号控制抓手

续表

序号	名称	功　　能	对应参数	出厂值(端子号)
33	控制抓手的输入信号范围	设置“控制抓手的输入信号范围”	HANDOUT	



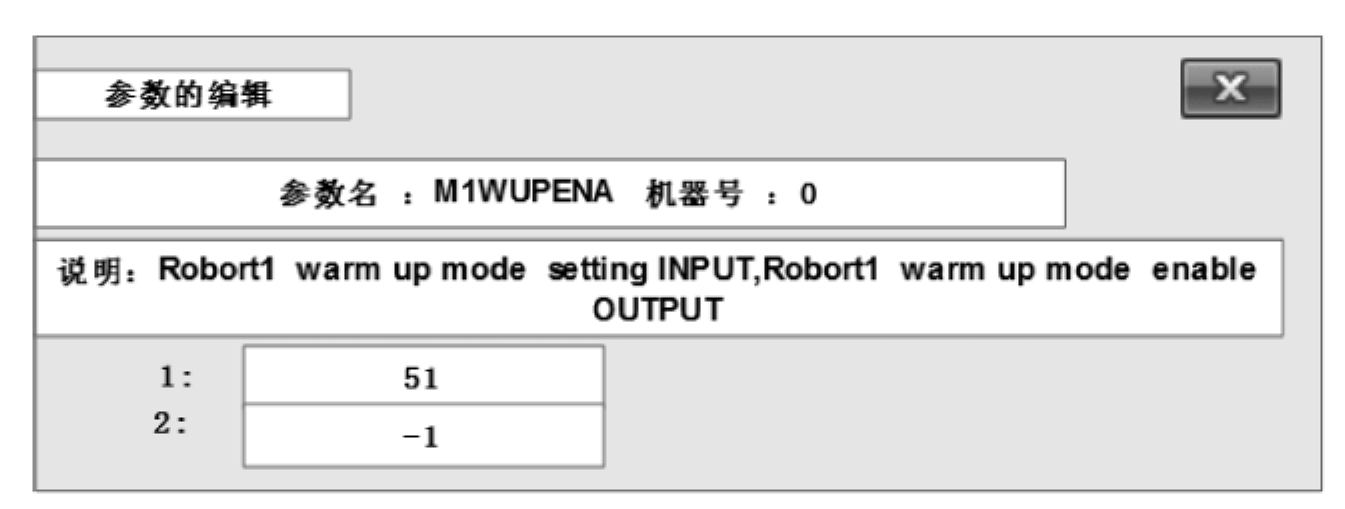
图中，设置对应本功能的输入端子号＝43～49，即输入端子43～49为“控制抓手的输入信号范围”

序号	名称	功　　能	对应参数	出厂值(端子号)
34	第N台机器人的抓手报警 N＝1～3	发出“第N台机器人抓手报警信号”	HNDERRn	

参数的编辑
参数名 ：HANDERR1　机器号 ：0
说明：Robort1　hand error requirement　During robort1 hang errr OUTPUT
1: 50
2: -1

图中，设置对应本功能的输入端子号＝50，如输入端子50＝ON，则发出“第N台机器人抓手报警信号”

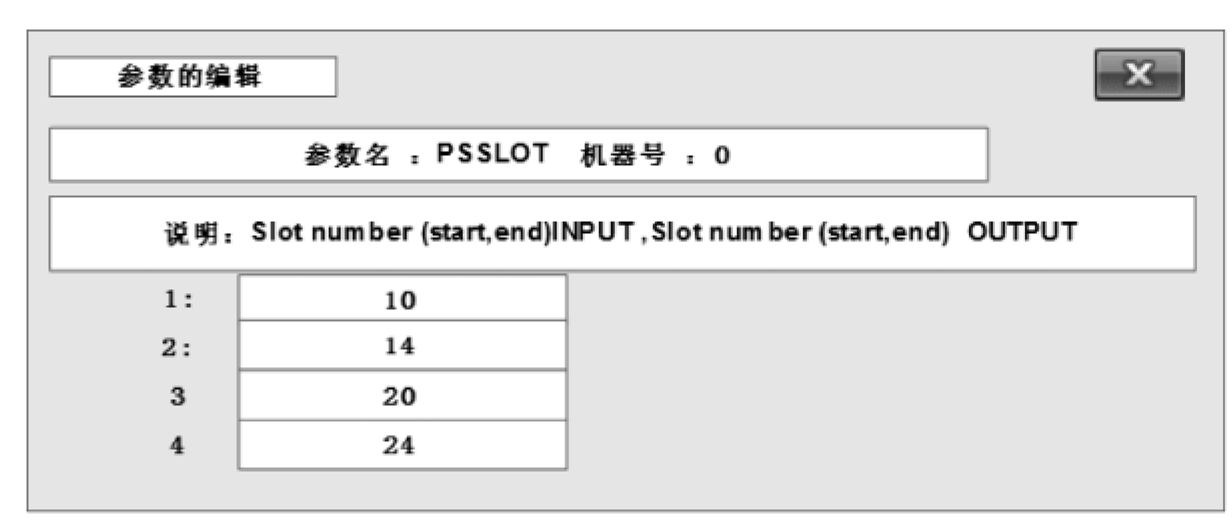
序号	名称	功　　能	对应参数	出厂值(端子号)
35	第N台机器人的气压报警 (N＝1～5)	发出“第N台机器人的气压报警”信号	AIRERRn	
36	第N台机器人预热运行模式有效	发出“第N台机器人预热运行模式有效”信号	MnWUPENA (N＝1～3)	



图中，设置对应本功能的输入端子号＝51，如输入端子51＝ON，则发出“第N台机器人预热运行模式有效”信号

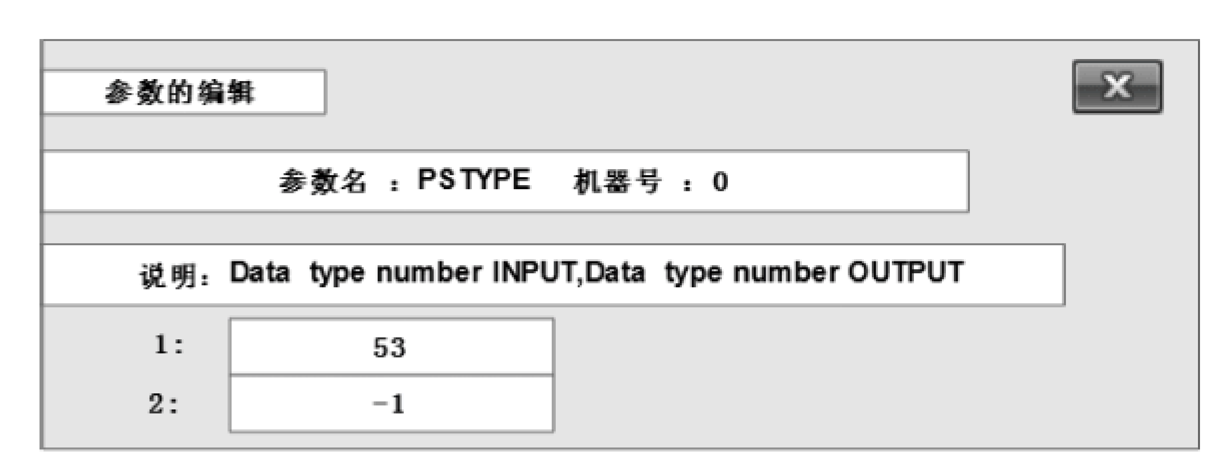
续表

序号	名称	功　　能	对应参数	出厂值(端子号)
37	指定需要输出位置数据的“任务区”号	指定需要输出位置数据的“任务区”号	PSSLOT	



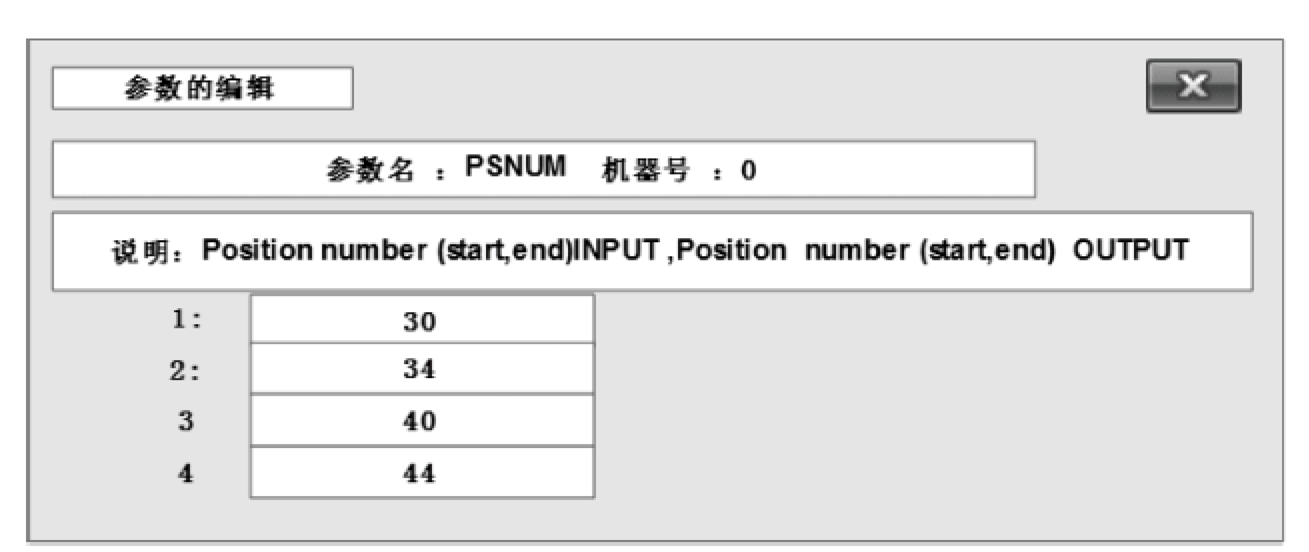
图中，设置对应本功能的输入端子号＝10～14，即输入端子 10～14 构成的数据为“需要输出位置数据的‘任务区’号”

38	位置数据类型	指定位置数据类型 1/0＝关节型变量/直交型变量	PSTYPE	



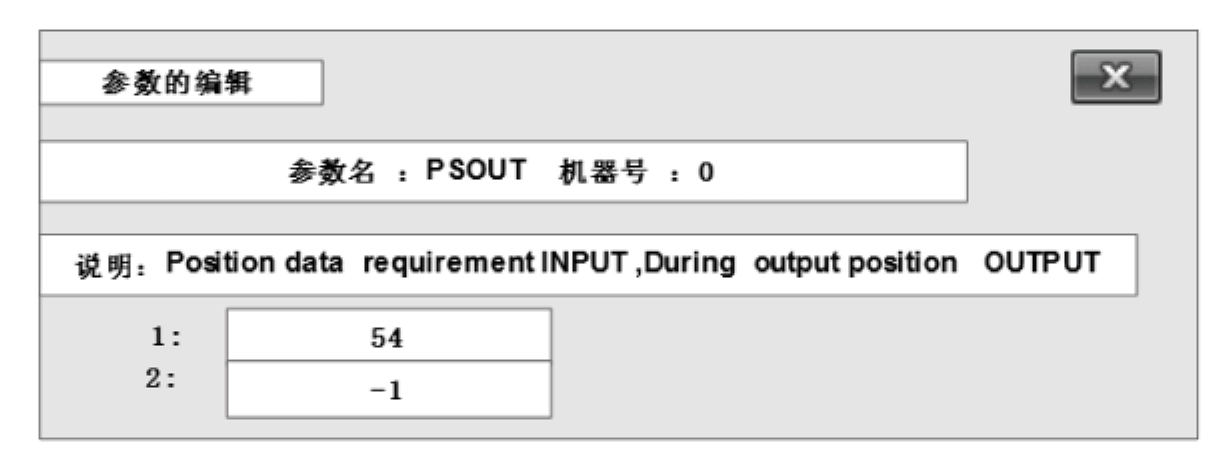
图中，设置对应本功能的输入端子号＝53，输入端子 53＝1/0，对应“关节型变量/直交型变量”

39	指定用一组数据表示“位置变量号”	指定用一组数据表示“位置变量号”	PSNUM	



图中，设置对应本功能的输入端子号＝30～34，即输入端子 30～34 构成的数据表示“位置变量号”

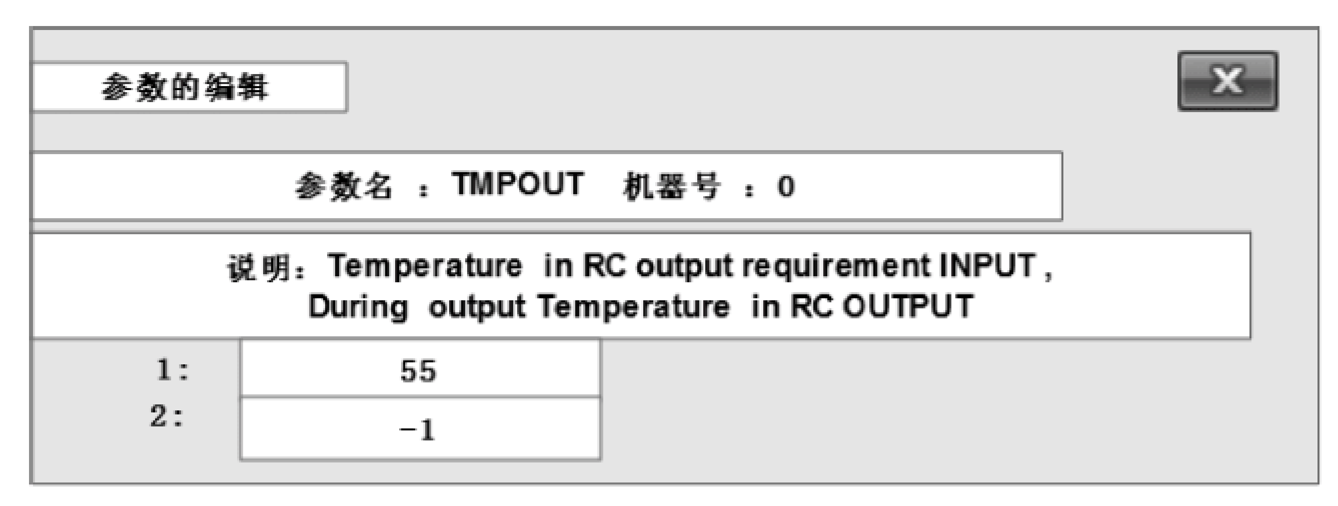
40	输出位置数据指令	指令输出当前“位置数据”	PSOUT	



图中，设置对应本功能的输入端子号＝54，如输入端子 54＝ON，则指令输出当前“位置数据”

续表

序号	名称	功　　能	对应参数	出厂值(端子号)
41	输出控制柜温度	指令输出控制柜实际温度	TMPOUT	



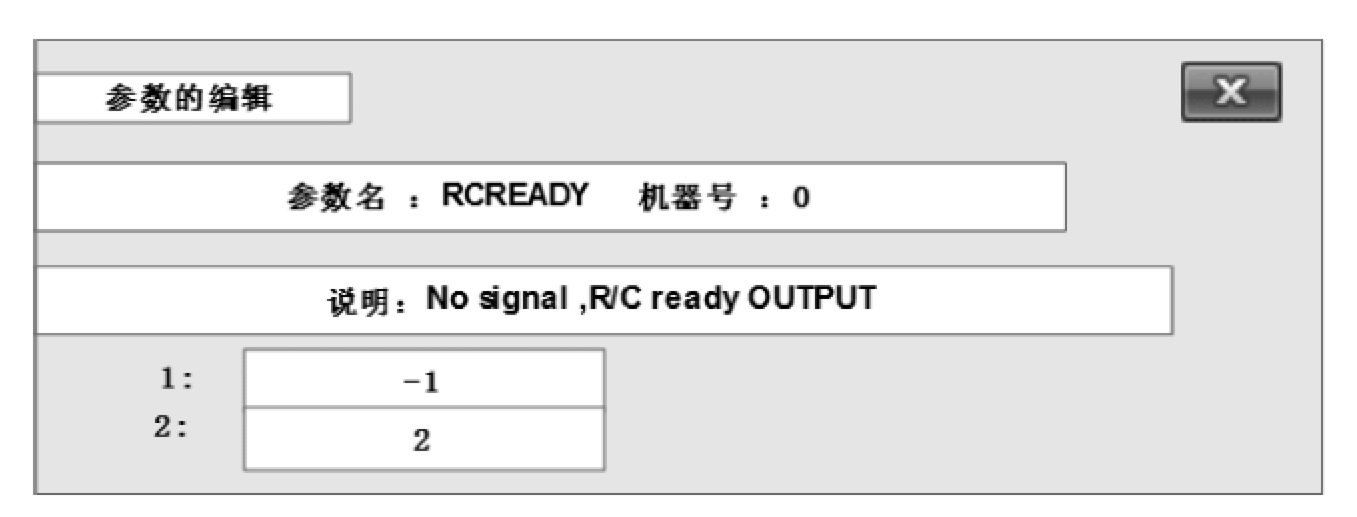
图中，设置对应本功能的输入端子号=55，如输入端子55=ON，则指令输出控制柜温度

5.3.3　专用输出信号详解

本节解释专用输出信号以及这些信号对应的参数，如表5-4所示。出厂值是指出厂时预分配的输出端子序号。由于同一参数包含输入信号与输出信号的内容，因此必须理解：参数只是表示某一功能，输入信号是驱动这一功能生效，输出信号是表示这一功能已经生效。

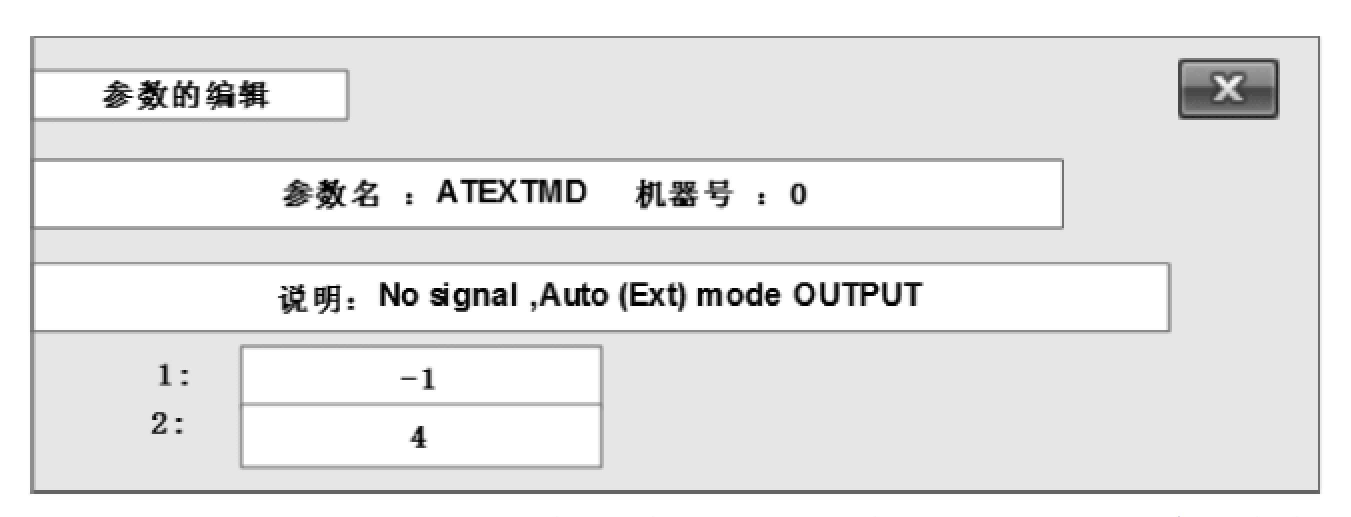
表5-4　专用输出信号

序号	名称	功　　能	对应参数	出厂值(端子号)
1	控制器电源ON	表示控制器电源ON，可以正常工作	RCREADY	



图中，设置对应本功能的输出端子号=2，如果控制器电源ON，则输出端子2=ON

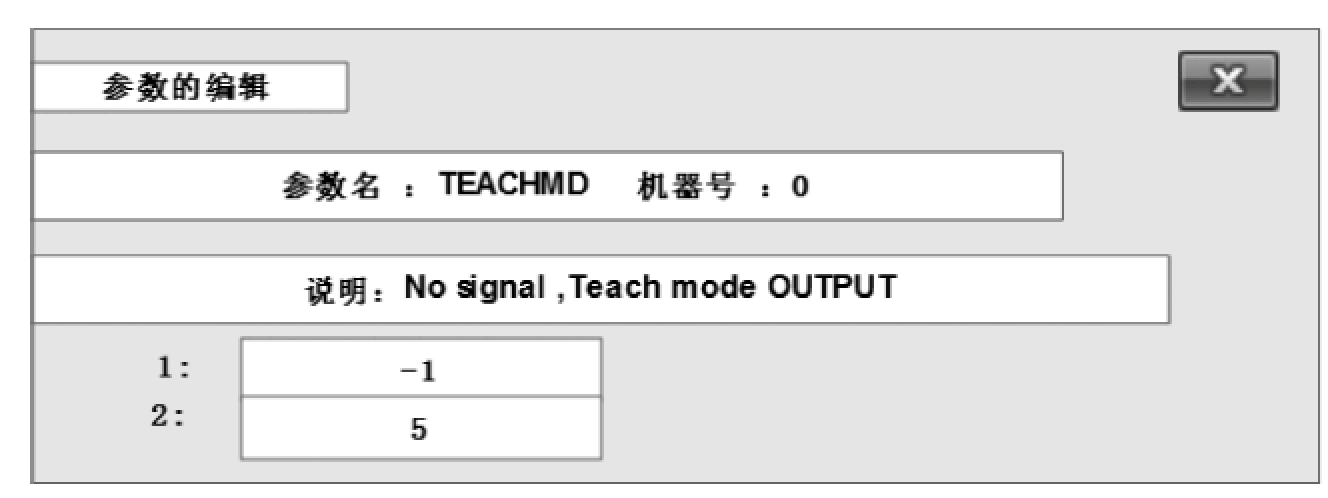
序号	名称	功　　能	对应参数	出厂值(端子号)
2	远程模式	表示操作面板选择自动模式，外部I/O信号操作有效	ATEXTMD	



图中，设置对应本功能的输出端子号=4，如果本功能生效，则输出端子4=ON

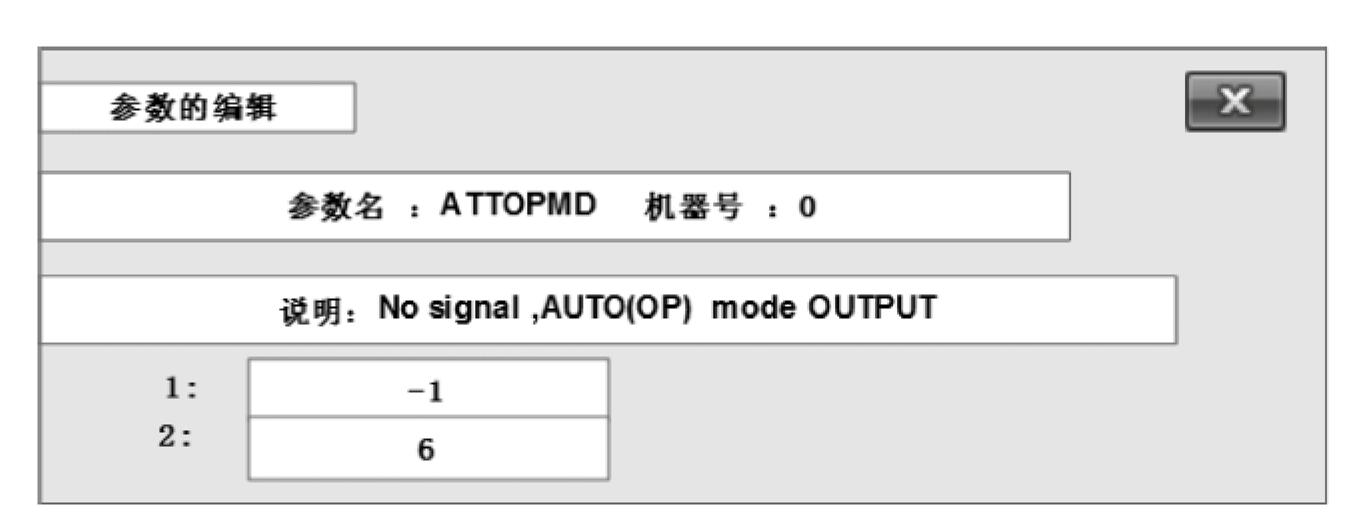
续表

序号	名称	功　　能	对应参数	出厂值(端子号)
3	示教模式	表示当前工作模式为“示教模式”	TEACHMD	



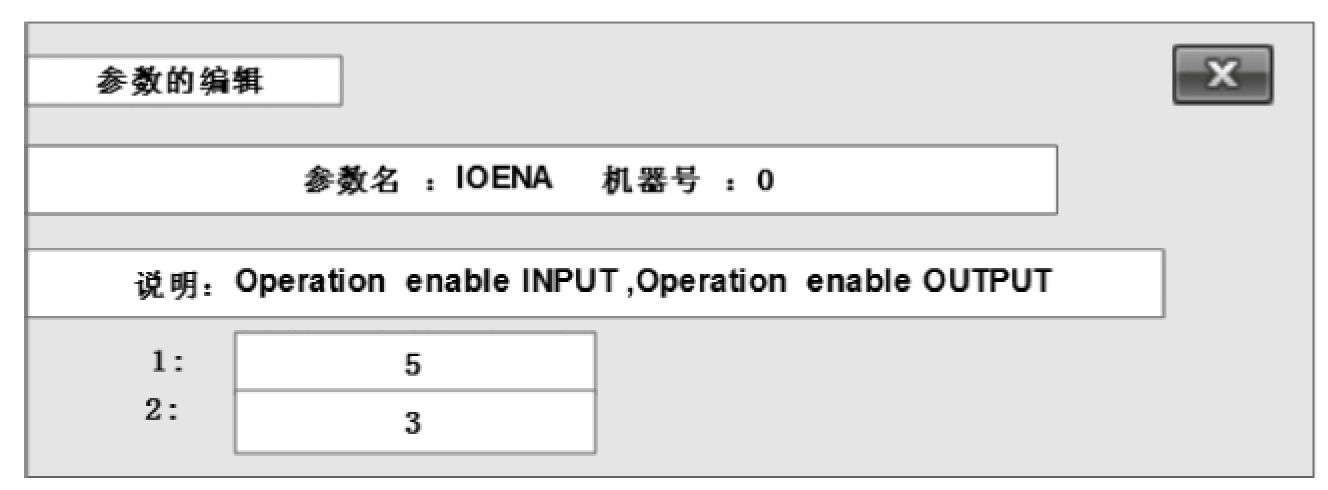
图中，设置对应本功能的输出端子号＝5，如果当前工作模式为“示教模式”，则输出端子 5＝ON

4	自动模式	表示当前工作模式为“自动模式”	ATTOPMD	



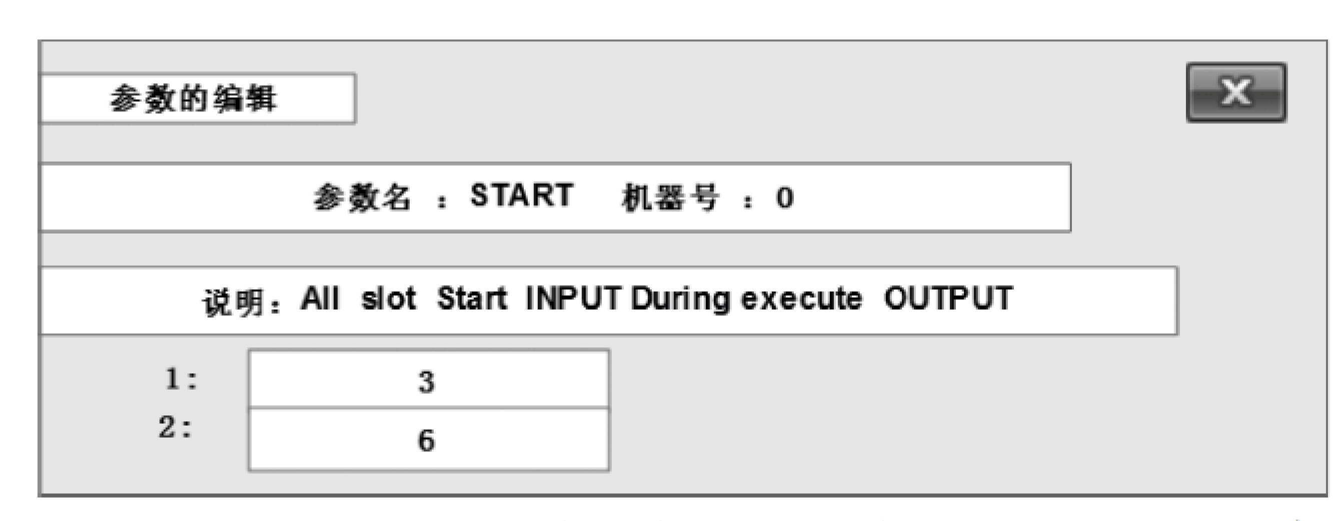
图中，设置对应本功能的输出端子号＝6，如果当前工作模式为“自动模式”，则输出端子 6＝ON

5	外部信号操作权有效	表示“外部信号操作权有效”。当输出 3＝ON，表示外部操作权有效	IOENA	3



图中，设置对应本功能的输出端子号＝3，如果外部操作权已经有效，则输出端子 3＝ON

6	程序已启动	表示机器人进入“程序已启动”状态	START	



图中，设置对应本功能的输出端子号＝6，如果机器人进入“程序已启动”状态，则输出端子 6＝ON

续表

序号	名称	功　　能	对应参数	出厂值(端子号)
7	程序停止	表示机器人进入“程序暂停”状态	STOP	

参数的编辑

参数名 ：STOP　机器号 ：0

说明：All slot Stop INPUT(no change) ,During wait OUTPUT

1: 0

2: 7

图中,设置对应本功能的输出端子号=7,如果机器人进入“程序暂停”状态,则输出端子 7=ON

序号	名称	功　　能	对应参数	出厂值(端子号)
8	程序停止	表示“程序暂停”状态	STOP2	

参数的编辑

参数名 ：STOP2　机器号 ：0

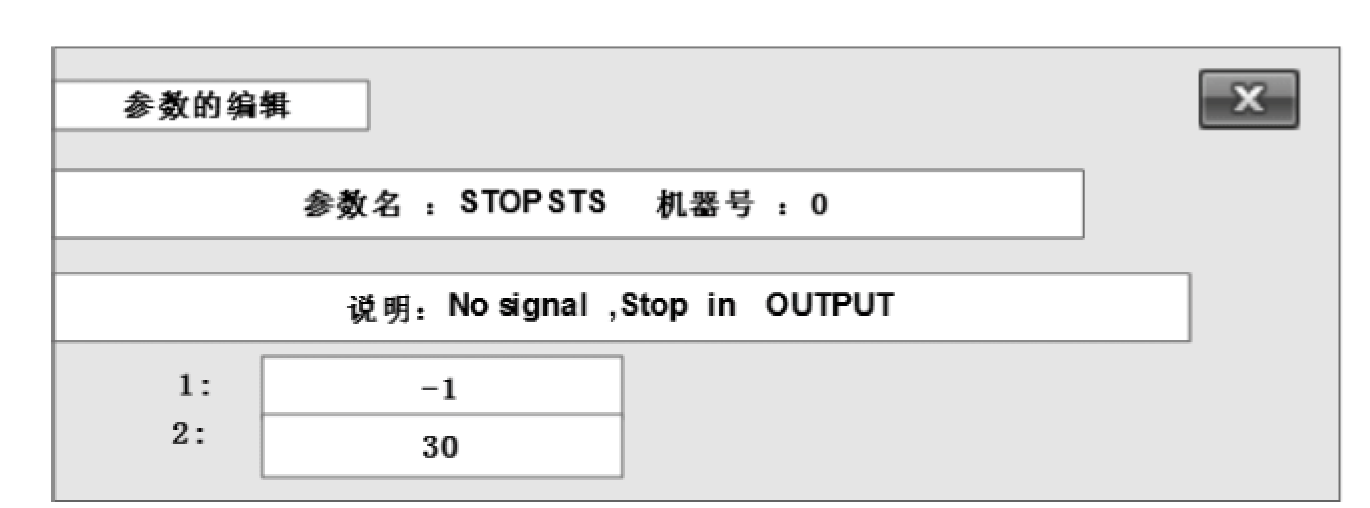
说明：All slot Stop INPUT,During wait OUTPUT

1: -1

2: 8

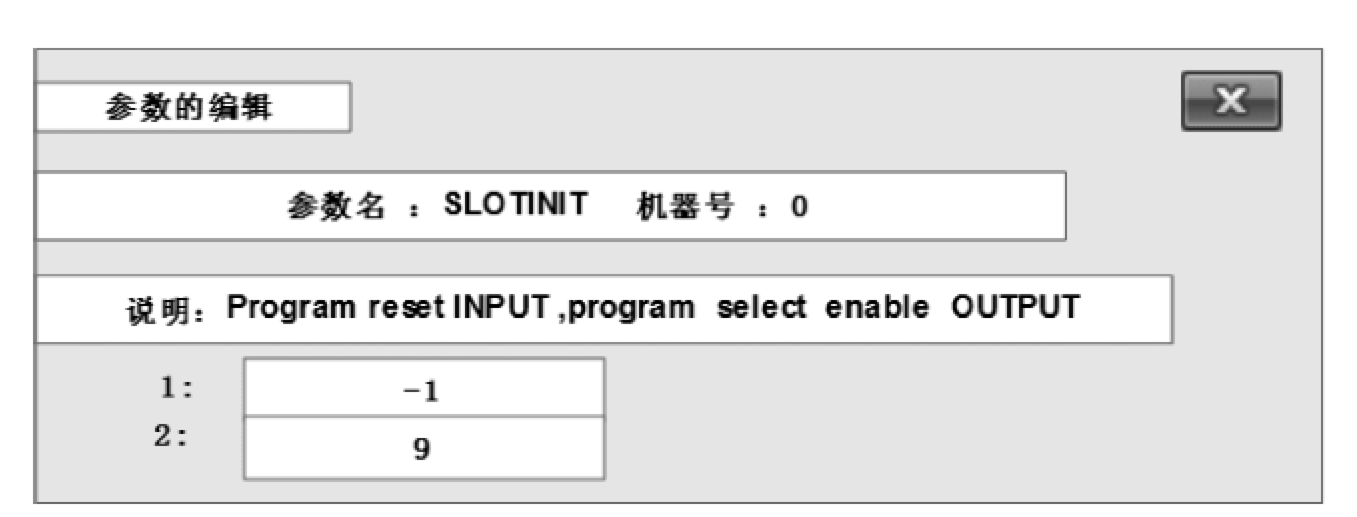
图中,设置对应本功能的输出端子号=8,如果机器人进入“程序暂停 2”状态,则输出端子 8=ON

序号	名称	功　　能	对应参数	出厂值(端子号)
9	STOP 信号输入	表示正在输入 STOP 信号	STOPSTS	



图中,设置对应本功能的输出端子号=30,如果正在输入 STOP 信号,则输出端子 30=ON

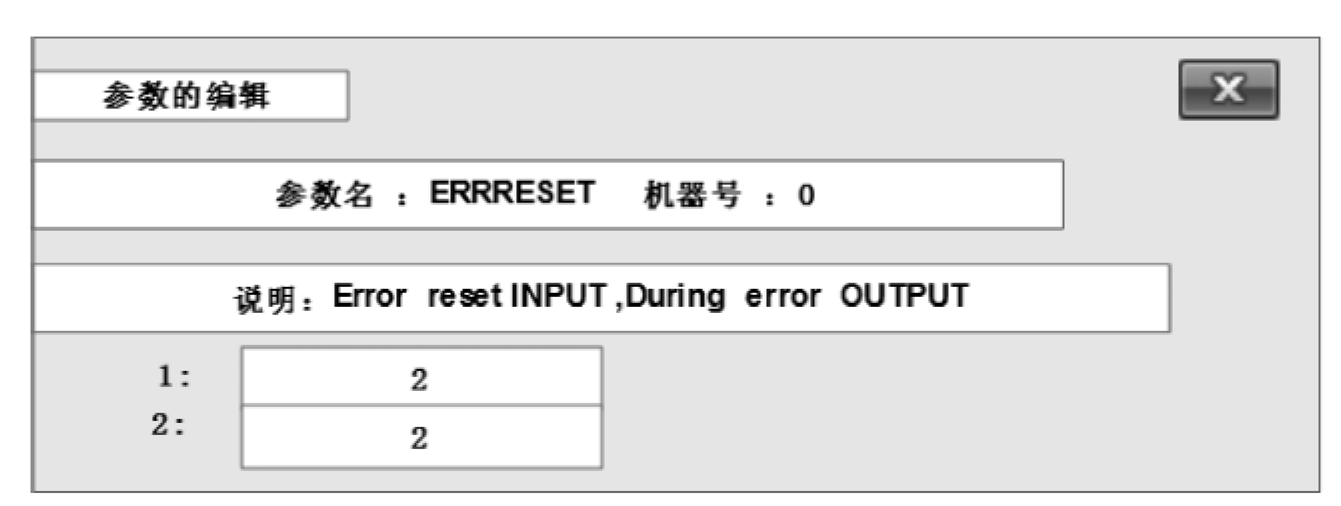
序号	名称	功　　能	对应参数	出厂值(端子号)
10	任务区中的程序可选择状态	表示“任务区处于程序可选择状态”	SLOTINIT	



图中,设置对应本功能的输出端子号=9,如果“任务区处于程序可选择状态”,则输出端子 9=ON

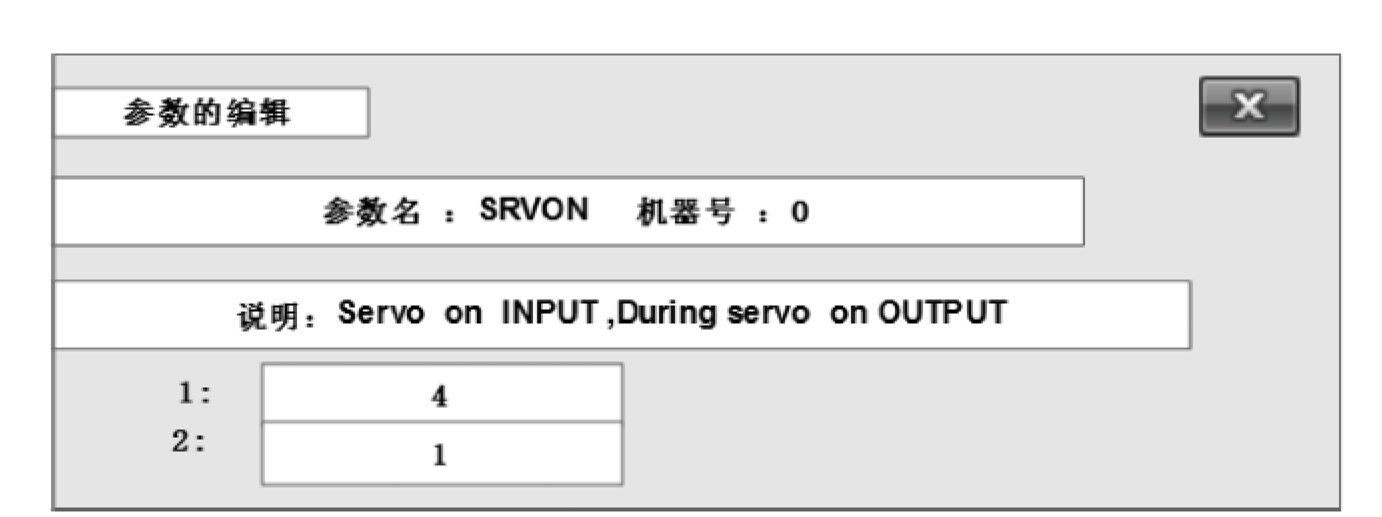
续表

序号	名称	功　　能	对应参数	出厂值(端子号)
11	报警发生中	表示系统处于“报警发生中”	ERRRESET	



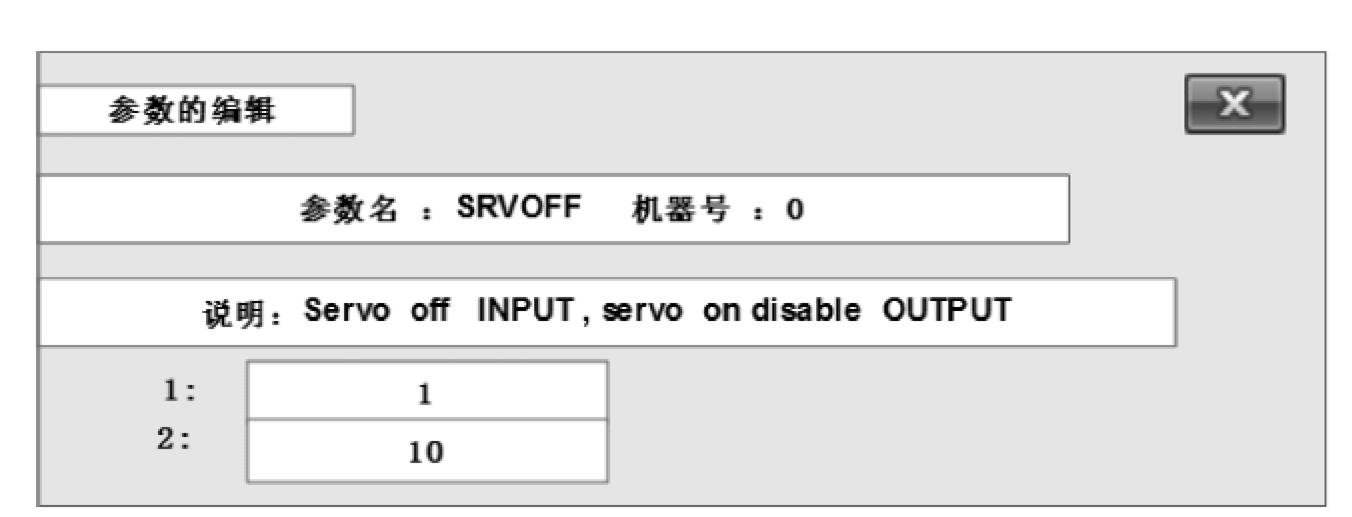
图中设置的输出端子号＝2,如系统处于“报警发生中”,则输出端子 2＝ON

12	伺服 ON	表示当前处于“伺服 ON”状态	SRVON	1



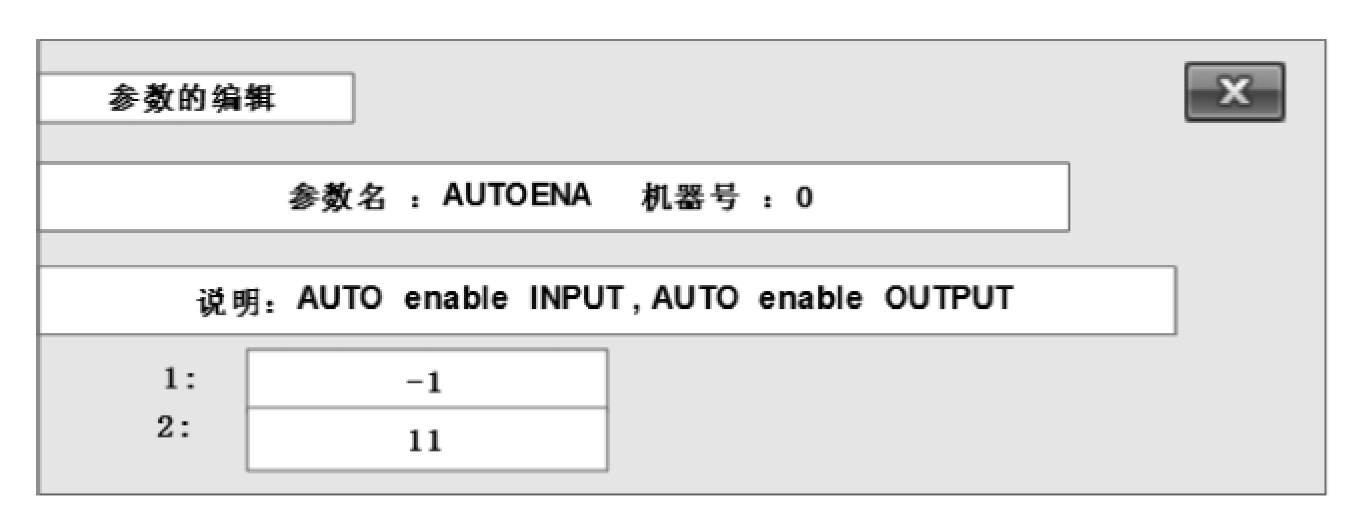
图中设置的输出端子号＝1,如果当前为“伺服 ON”状态,则输出端子 1＝ON

13	伺服 OFF	表示当前处于“伺服 OFF”状态	SRVOFF	



图中设置的输出端子号＝10,如果当前处于“伺服 OFF”状态,则输出端子 10＝ON

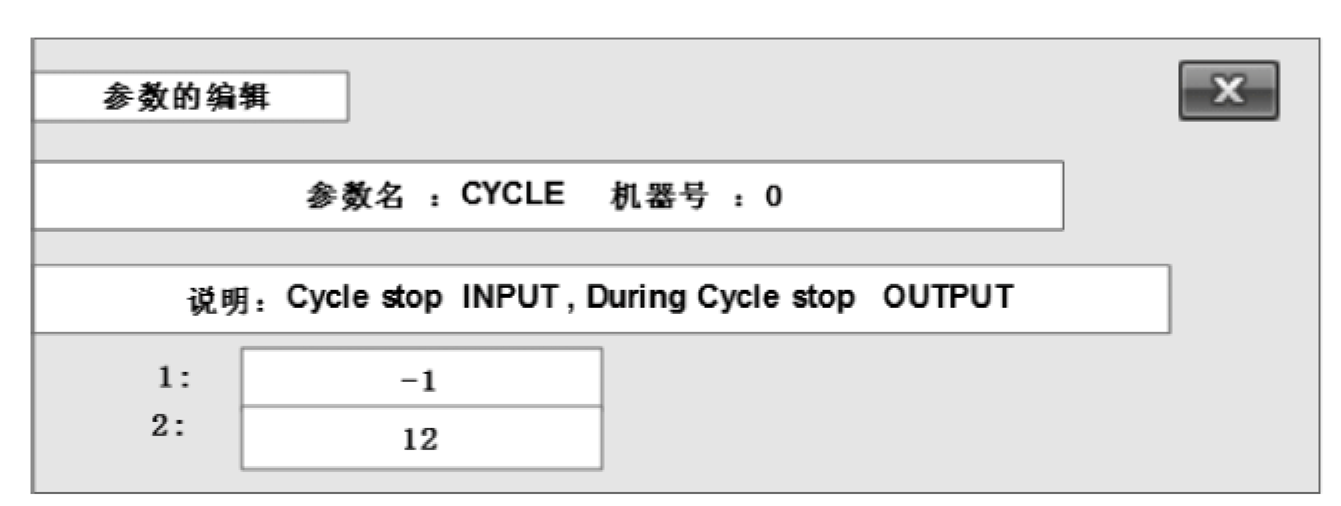
14	可自动运行	表示当前处于“可自动运行”状态	AUTOENA	



图中,设置对应本功能的输出端子号＝11,如果当前处于“可自动运行”状态,则输出端子 11＝ON

续表

序号	名称	功　　能	对应参数	出厂值(端子号)
15	循环停止信号	表示“循环停止信号”正输入中	CYCLE	



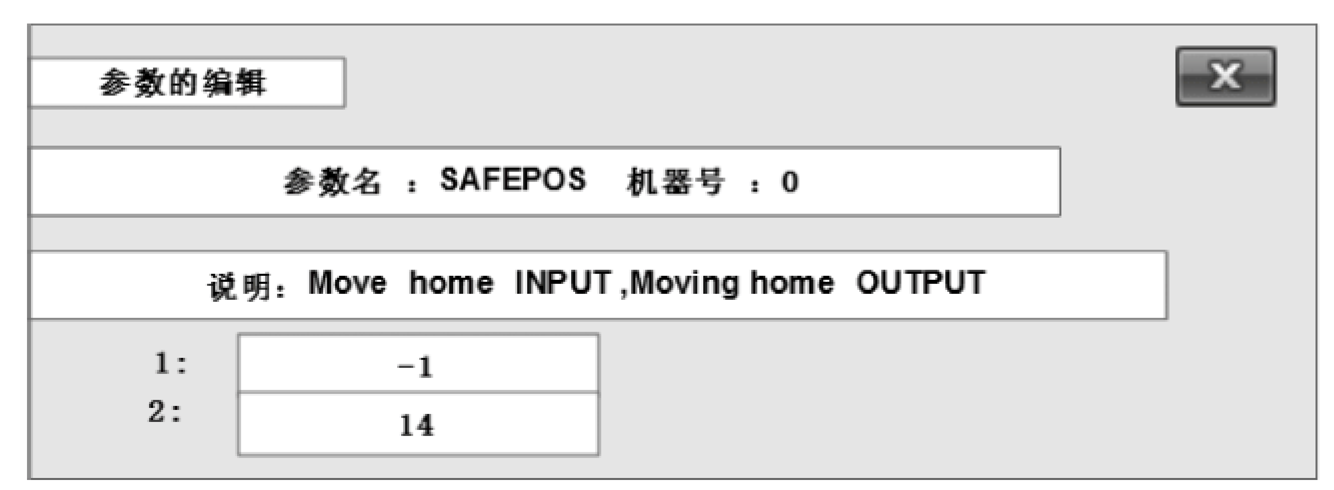
图中，设置对应本功能的输出端子号＝12，如果“循环停止信号”正输入中，则输出端子 12＝ON

16	机械锁定状态	表示机器人处于“机械锁定状态”。“机械锁定状态”是程序运行，机器人不动作	MELOCK	

参数的编辑
参数名 ：MELOCK　机器号 ：0
说明：Machine lock INPUT ,Machine lock OUTPUT
1:　-1
2:　13

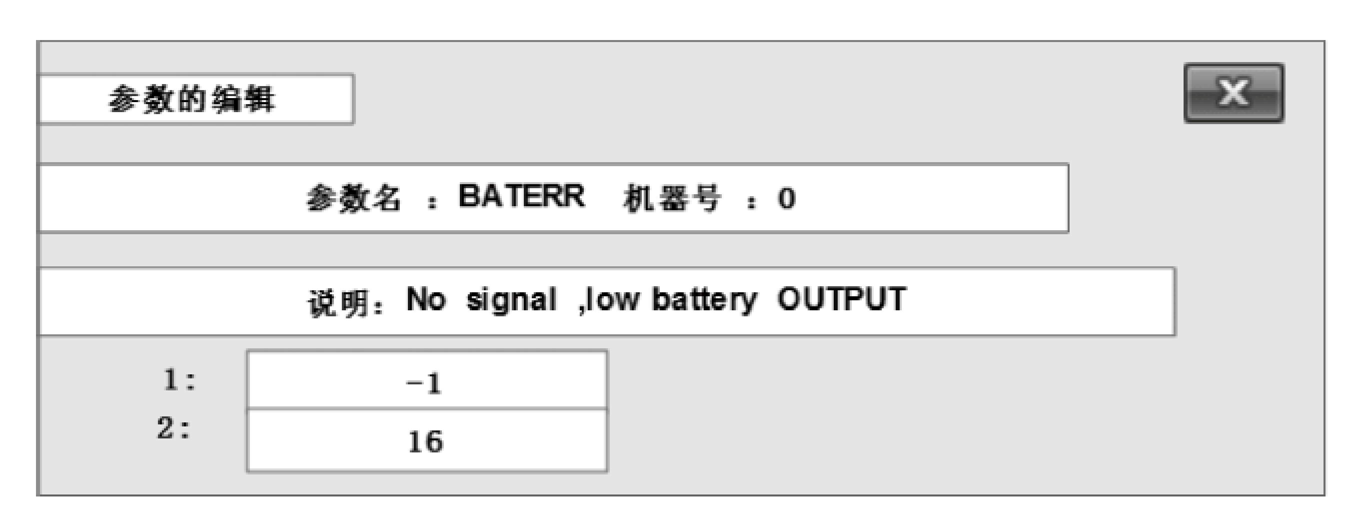
图中，设置对应本功能的输出端子号＝13，如果机器人处于“机械锁定状态”则输出端子 13＝ON

17	回归待避点状态	表示机器人处于“回归待避点状态”	SAFEPOS	



图中，设置对应本功能的输出端子号＝14，如果机器人处于“回归待避点状态”，则输出端子 14＝ON

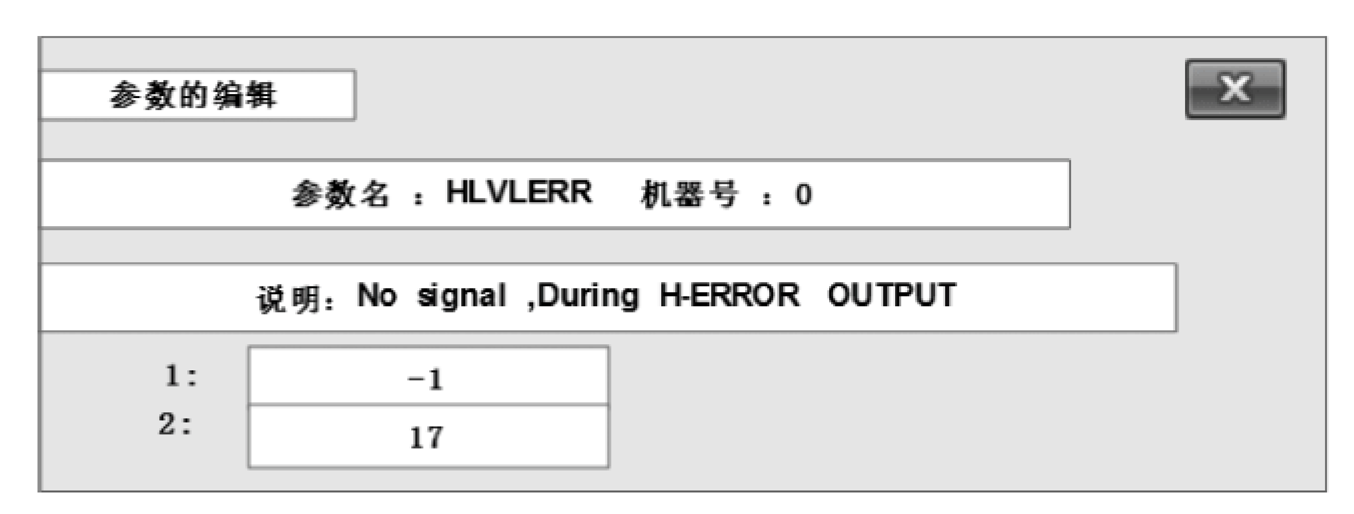
18	电池电压过低	表示机器人“电池电压过低”	BATERR	



图中，设置对应本功能的输出端子号＝16，如果机器人处于“电池电压过低状态”，则输出端子 16＝ON

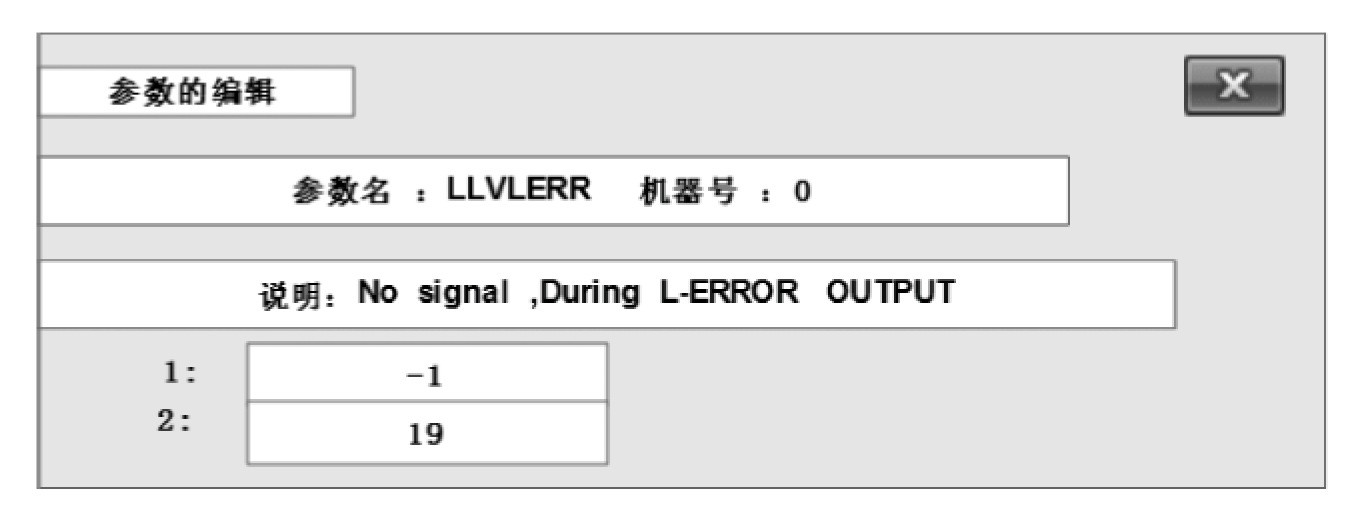
续表

序号	名称	功　　能	对应参数	出厂值(端子号)
19	严重级报警	表示机器人出现“严重级故障报警”	HLVLERR	



图中,设置对应本功能的输出端子号=17,如果机器人处于“严重级故障报警”,则输出端子 17=ON

序号	名称	功　　能	对应参数	出厂值(端子号)
20	轻微级故障报警	表示机器人出现“轻微级故障报警”	LLVLERR	



图中,设置对应本功能的输出端子号=19,如果机器人处于“轻微级故障报警”,则输出端子 19=ON

序号	名称	功　　能	对应参数	出厂值(端子号)
21	警告型故障	表示机器人出现“警告型故障”	CLVLERR	
22	机器人急停	表示机器人处于“急停状态”	EMGERR	

参数的编辑
参数名 ：CLVLERR 机器号 ：0
说明：No signal ,During Caution OUTPUT
1： -1
2： 20

图中,设置对应本功能的输出端子号=20,如果机器人处于“急停状态”,则输出端子 20=ON

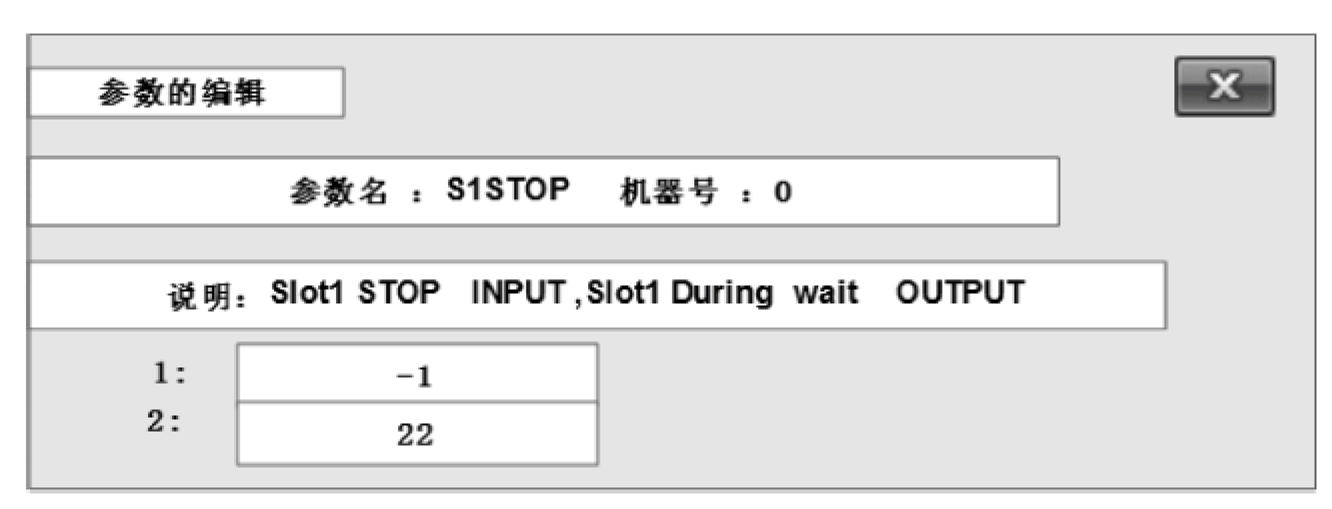
序号	名称	功　　能	对应参数	出厂值(端子号)
23	第 N 任务区程序在运行中	表示“第 N 任务区程序在运行中”	SnSTART	

参数的编辑
参数名 ：S1START 机器号 ：0
说明：Slot1 start INPUT ,Slot1 During Execute OUTPUT
1： -1
2： 21

图中,设置对应本功能的输出端子号=21,如果机器人处于“第 1 任务区程序运行状态”,则输出端子 21=ON

续表

序号	名称	功　　能	对应参数	出厂值(端子号)
24	第N任务区程序在暂停中	表示“第N任务区程序在暂停中”	SnSTOP	



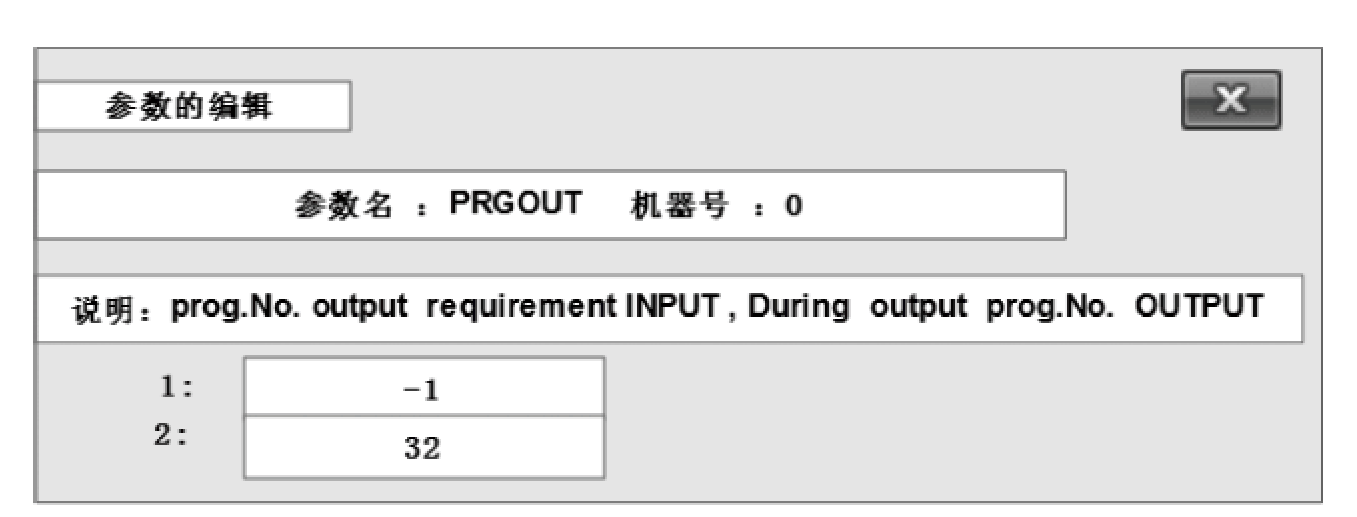
图中，设置对应本功能的输出端子号=22，如果机器人处于“第1任务区程序暂停中状态”，则输出端子22=ON

25	第N台机器人伺服OFF	表示“第N台机器人伺服OFF”	SnSRVOFF	
26	第N台机器人伺服ON	表示“第N台机器人伺服ON”	SnSRVON	
27	第N台机器人机械锁定	表示“第N台机器人处于机械锁定”状态	SnMELOCK	
28	数据输出地址号	对应于数据输出，指定输出信号的“起始位”“结束位”	IODATA	

参数的编辑
参数名 ： IODATA　机器号 ： 0
说明： value input signal (start,end)INPUT ,value output signal (start,end) OUTPUT
1： -1
2： -1
3　24
4　31

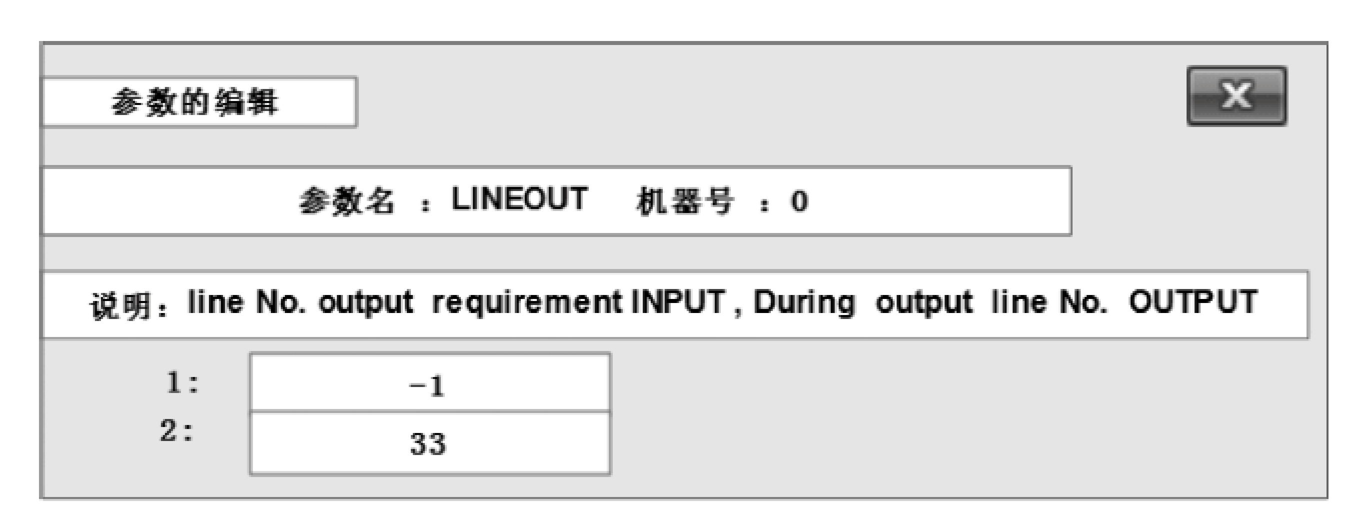
图中，设置对应本功能的输出端子号=24～31，则输出端子24～31的ON/OFF状态构成了一组数据

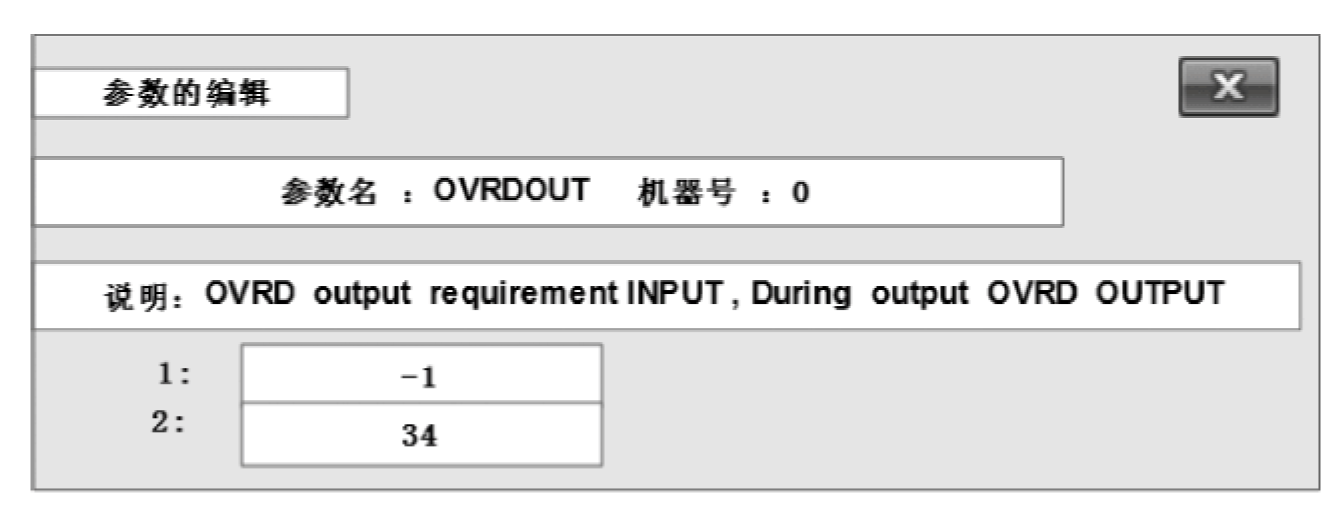
29	“程序号”数据输出中	表示当前正在输出“程序号”	PRGOUT	

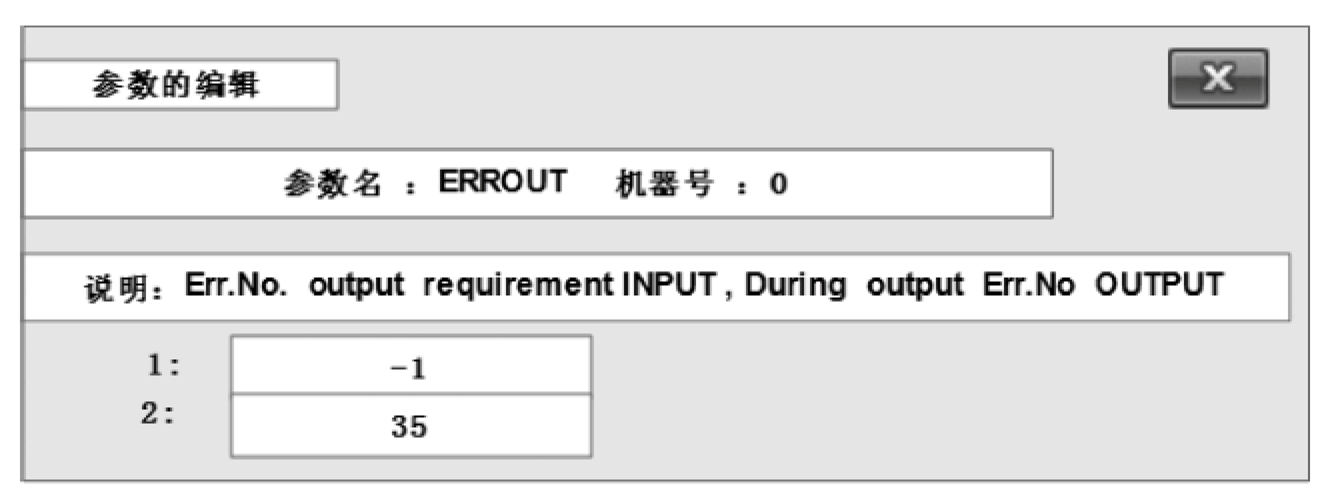


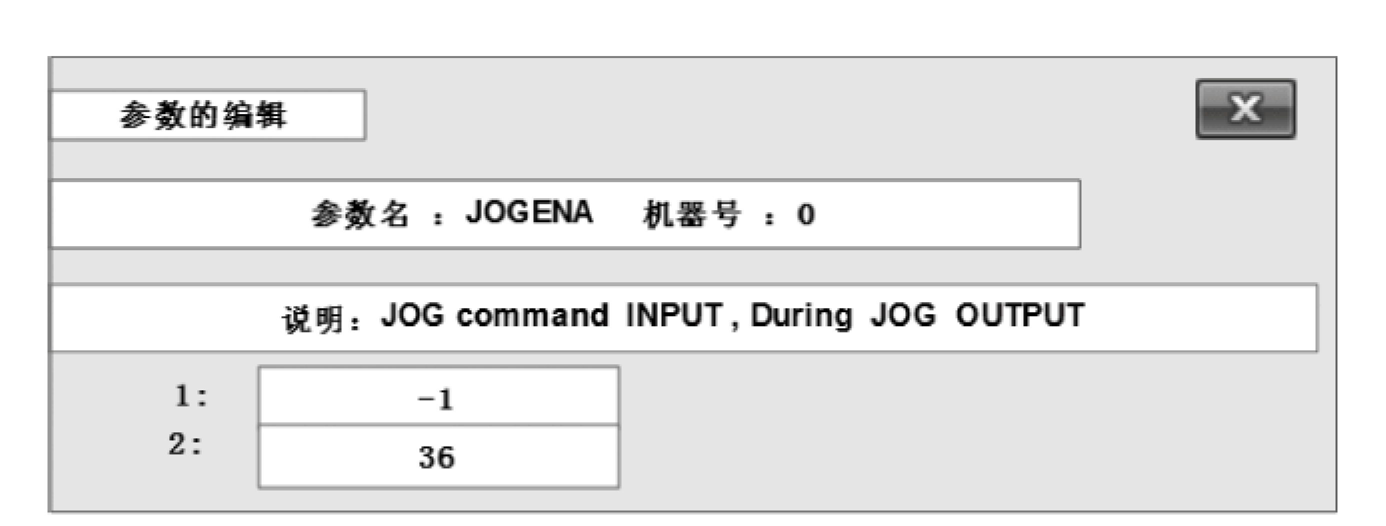
图中，设置对应本功能的输出端子号=32，如果机器人当前正在输出“程序号”，则输出端子32=ON

续表

<table>
<tr><th>序号</th><th>名称</th><th>功　　能</th><th>对应参数</th><th>出厂值(端子号)</th></tr>
<tr><td>30</td><td>“程序行号”数据输出中</td><td>表示当前正在输出“程序行号”</td><td>LINEOUT</td><td></td></tr>
<tr><td colspan="5">

图中，设置对应本功能的输出端子号＝33，如果机器人当前正在输出“程序行号”，则输出端子 33＝ON</td></tr>
<tr><td>31</td><td>“速度比例”数据输出中</td><td>表示当前正在输出“速度比例”</td><td>OVRDOUT</td><td></td></tr>
<tr><td colspan="5">

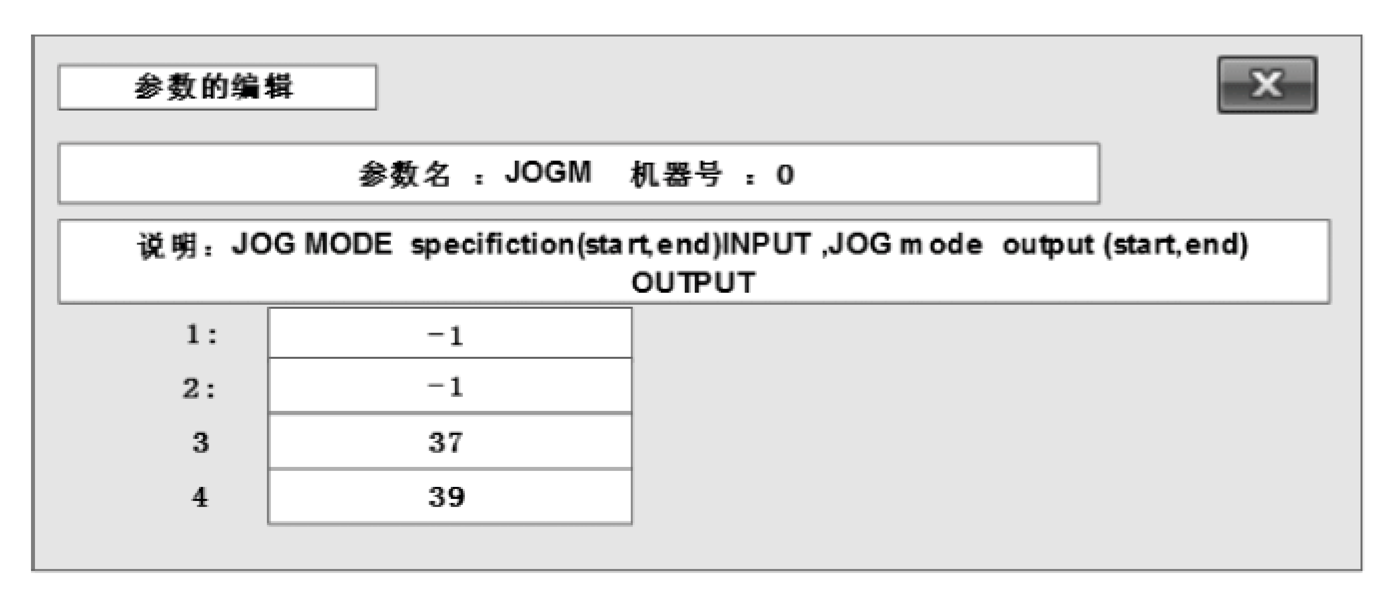
图中，设置对应本功能的输出端子号＝34，如果机器人当前正在输出“速度比例”，则输出端子 34＝ON</td></tr>
<tr><td>32</td><td>“报警号”输出中</td><td>表示当前正在输出“报警号”</td><td>ERROUT</td><td></td></tr>
<tr><td colspan="5">

图中，设置对应本功能的输出端子号＝35，如果机器人当前正在输出“报警号”，则输出端子 35＝ON</td></tr>
<tr><td>33</td><td>JOG 有效状态</td><td>表示当前处于“JOG 有效状态”</td><td>JOGENA</td><td></td></tr>
<tr><td colspan="5">

图中，设置对应本功能的输出端子号＝36，如果机器人当前处于“JOG 有效状态”，则输出端子 36＝ON</td></tr>
</table>

续表

序号	名称	功　　能	对应参数	出厂值(端子号)
34	JOG模式	表示当前的“JOG模式”	JOGM	



图中，设置对应本功能的输出端子号＝37～39，输出端子37～39构成的数据表示了JOG的工作模式

序号	名称	功　　能	对应参数	出厂值(端子号)
35	JOG报警无效状态	JOG报警无效状态	JOGNER	

参数的编辑

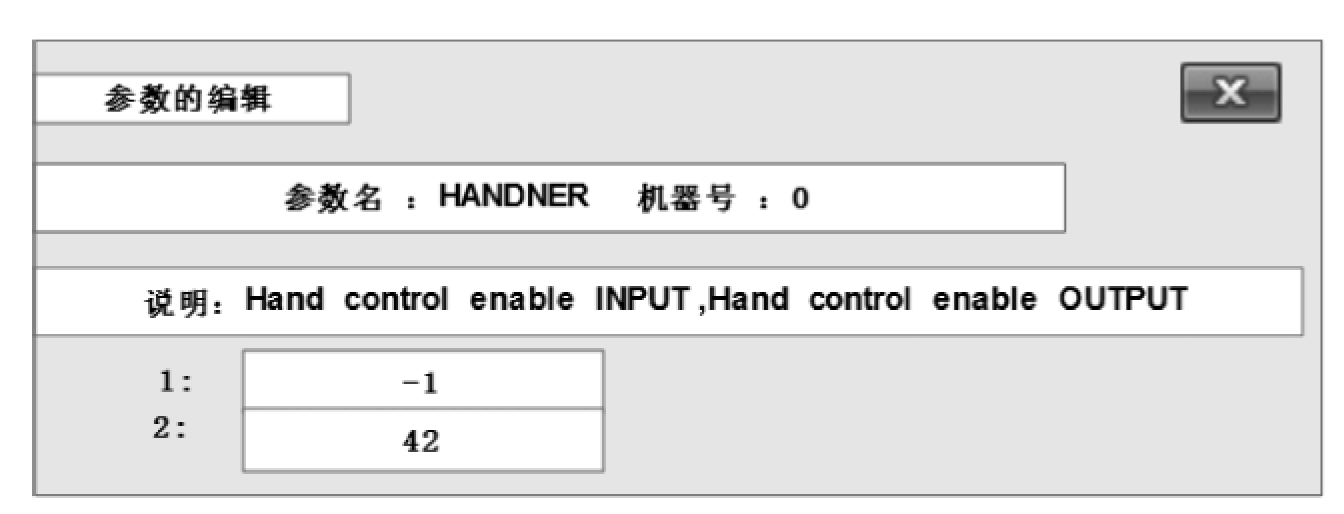
参数名 ：ERROUT　机器号 ：0

说明：Err.No. output requirement INPUT , During output Err.No OUTPUT

1：	-1
2：	35

图中，设置对应本功能的输出端子号＝40，如果机器人当前处于“JOG报警无效状态”，则输出端子40＝ON

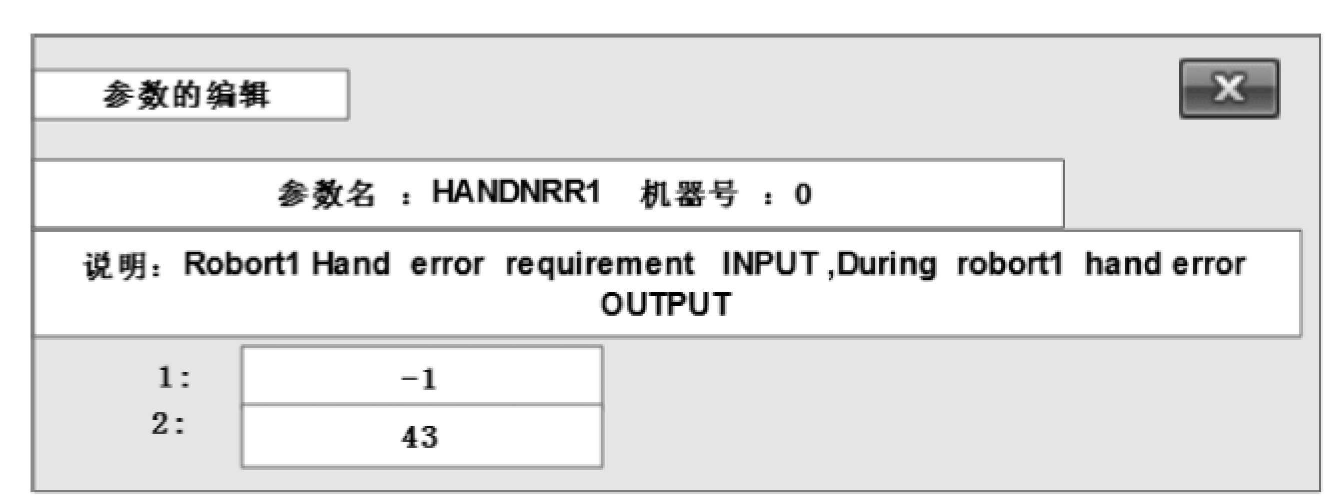
序号	名称	功　　能	对应参数	出厂值(端子号)
36	抓手工作状态	输出抓手工作状态 (输出信号部分)	HNDCNTLn	
37	抓手工作状态	输出抓手工作状态 (输入信号部分)	HNDSTSn	
38	外部信号对抓手控制的有效/无效状态	表示“外部信号对抓手控制的有效/无效状态”	HANDENA	



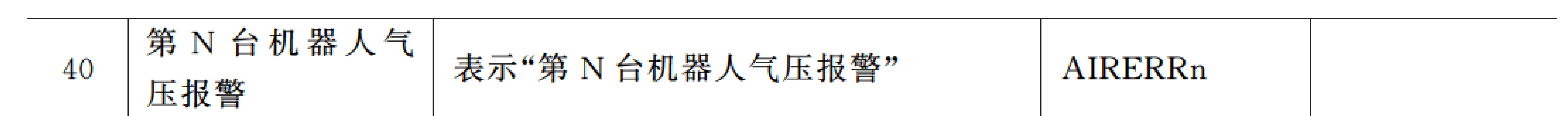
图中，设置对应本功能的输出端子号＝42，如果机器人当前处于“外部信号对抓手控制有效状态”，则输出端子42＝ON

续表

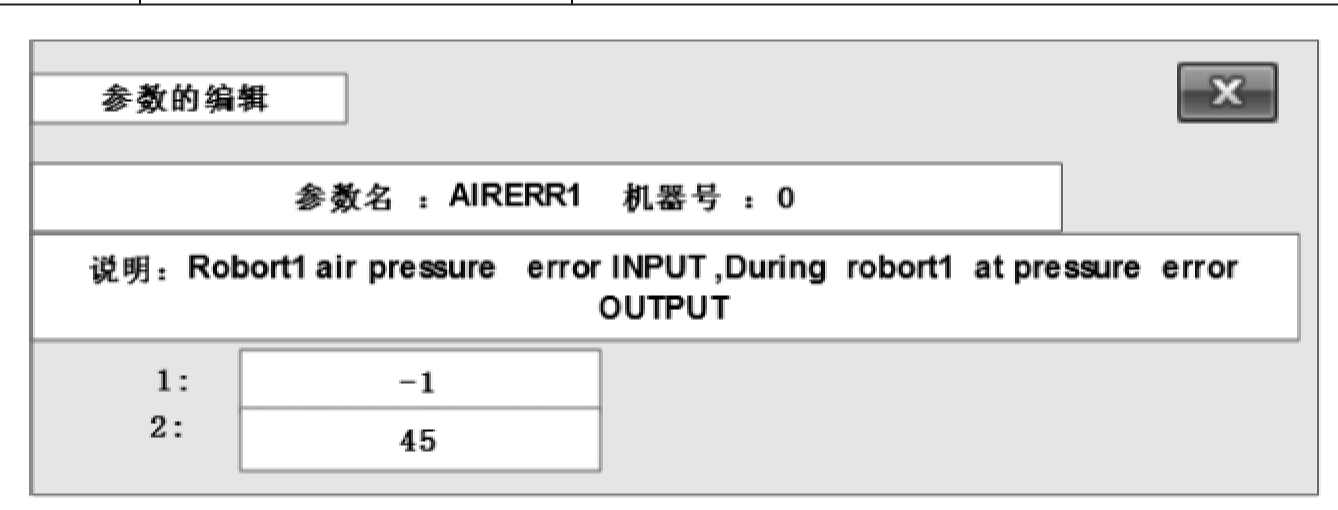
序号	名称	功　能	对应参数	出厂值(端子号)
39	第N台机器人抓手报警	表示“第N台机器人抓手报警”	HNDERRn	



图中，设置对应本功能的输出端子号＝43，如果1#机器人当前处于“抓手报警”，则输出端子43＝ON

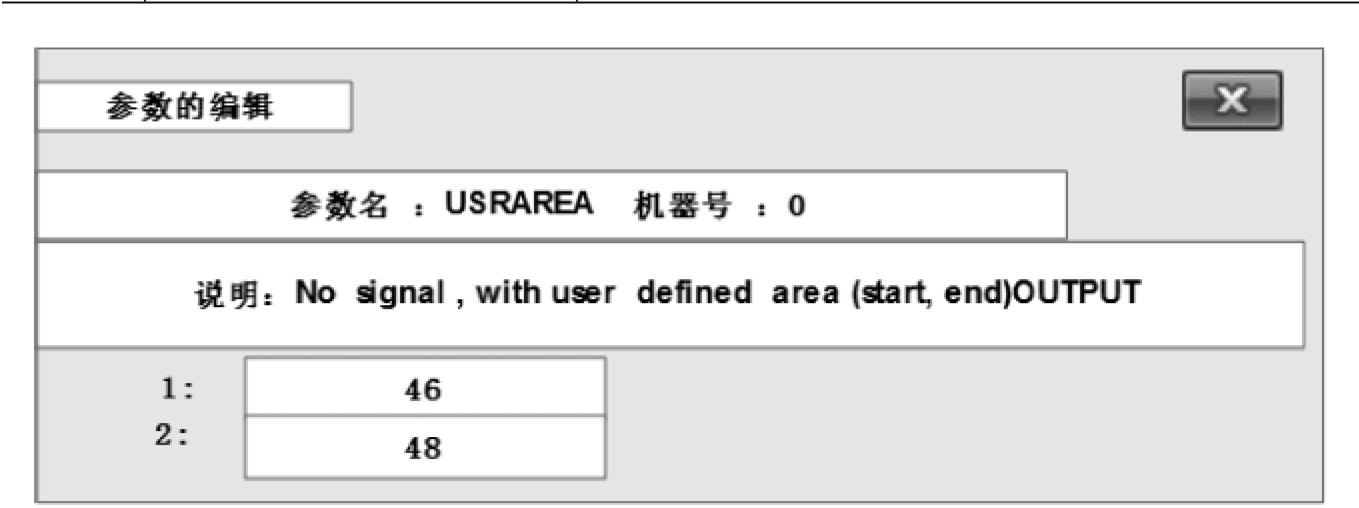


40	第N台机器人气压报警	表示“第N台机器人气压报警”	AIRERRn	



图中，设置对应本功能的输出端子号＝45，如果1#机器人当前处于“气压报警状态”，则输出端子45＝ON

41	用户定义区编号	用输出端子“起始位”“结束位”表示“用户定义区编号”	USRAREA	



图中，设置对应本功能的输出端子号＝46～48，输出端子46～48构成的数据表示了“用户定义区编号”

42	易损件维修时间	表示易损件到达“维修时间”	MnPTEXC	

参数的编辑

参数名 ：M1PTEXC　机器号 ：0

说明：No signal , robot1 warning which urges exchange of parts

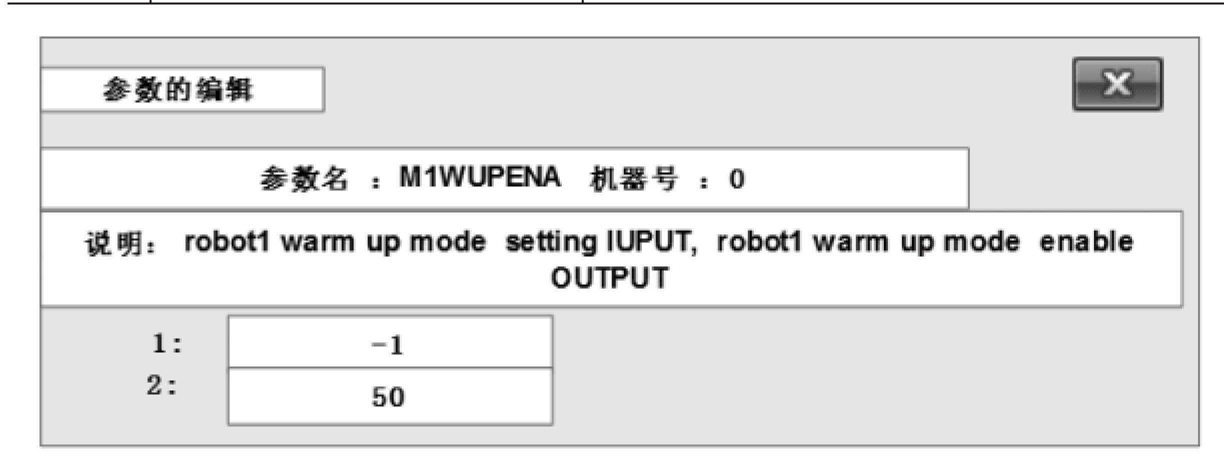
1：　-1

2：　49

图中，设置对应本功能的输出端子号＝49，如果机器人易损件到达“维修时间”，则输出端子49＝ON

续表

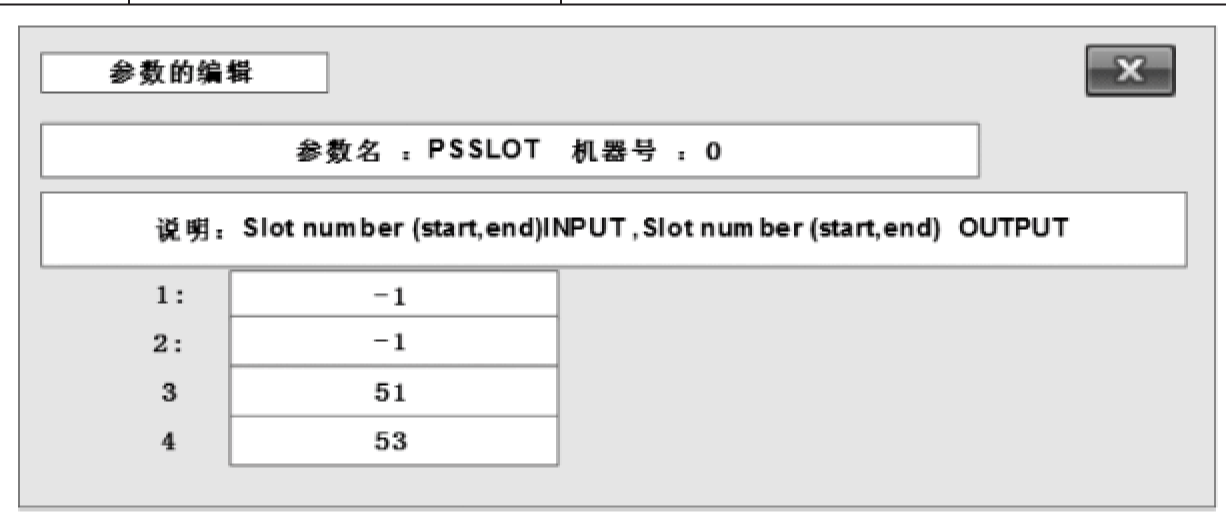
序号	名称	功　能	对应参数	出厂值(端子号)
43	机器人处于“预热工作模式”	表示“机器人处于预热工作模式”	MnWUPENA	


参数的编辑
参数名 ：M1WUPENA　机器号 ：0
说明： robot1 warm up mode setting IUPUT, robot1 warm up mode enable OUTPUT

1:	−1
2:	50

图中，设置对应本功能的输出端子号=50，如果机器人处于“预热工作模式”，则输出端子 50=ON

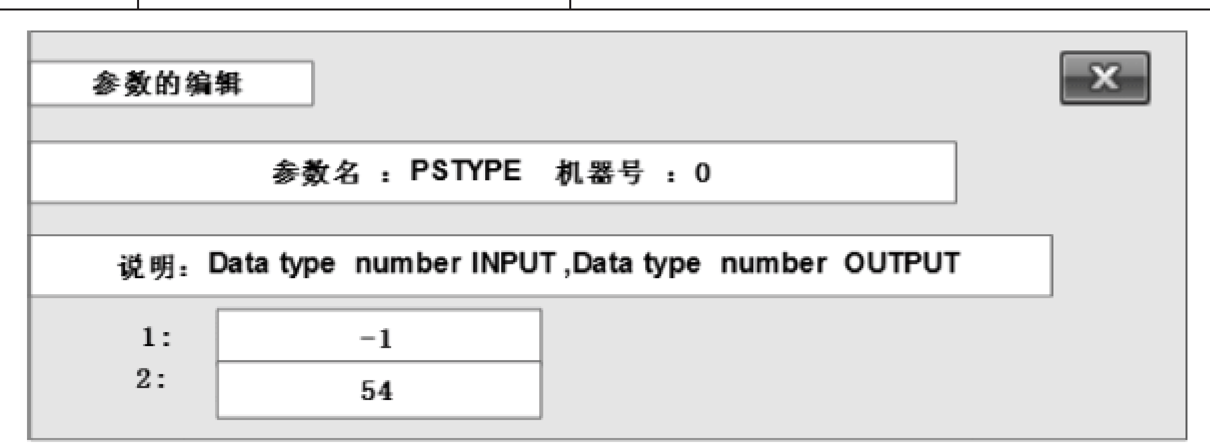
序号	名称	功　能	对应参数	出厂值(端子号)
44	输出位置数据的任务区编号	用输出端子“起始位”“结束位”表示“输出位置数据的任务区编号”	PSSLOT	


参数的编辑
参数名 ：PSSLOT　机器号 ：0
说明：Slot number (start,end)INPUT ,Slot number (start,end)　OUTPUT

1:	−1
2:	−1
3	51
4	53

图中，设置对应本功能的输出端子号=51～53，输出端子 51～53 构成的数据表示了“输出位置数据的任务区编号”

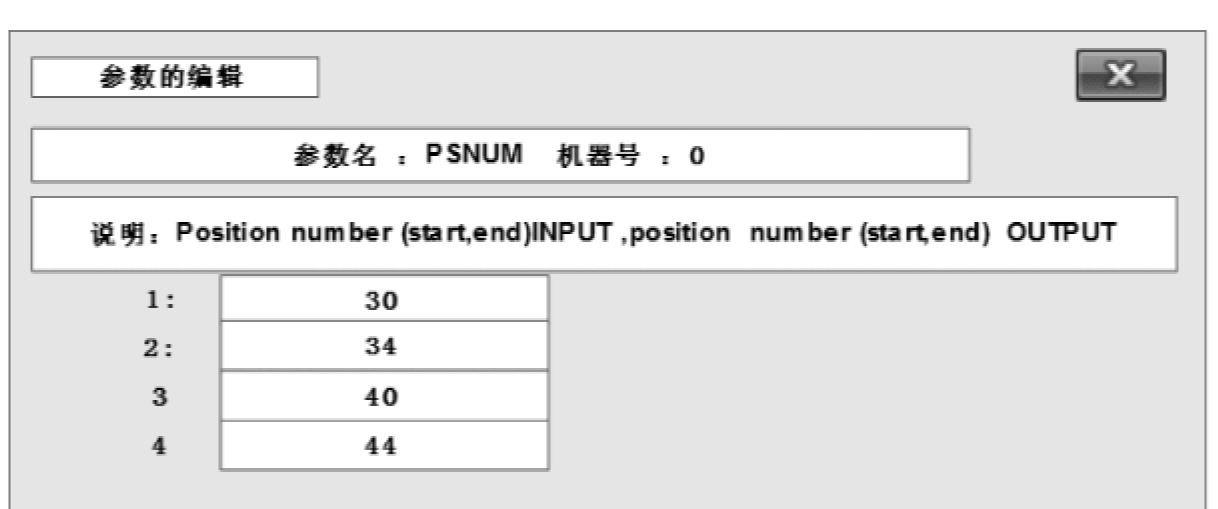
序号	名称	功　能	对应参数	出厂值(端子号)
45	输出的“位置数据类型”	表示输出的“位置数据类型”是关节型还是直交型	PSTYPE	


参数的编辑
参数名 ：PSTYPE　机器号 ：0
说明：Data type number INPUT ,Data type number OUTPUT

1:	−1
2:	54

图中，设置对应本功能的输出端子号=54，如果“位置数据类型=关节型”，则输出端子 54=ON。如果“位置数据类型=直交型”，则输出端子 54=OFF

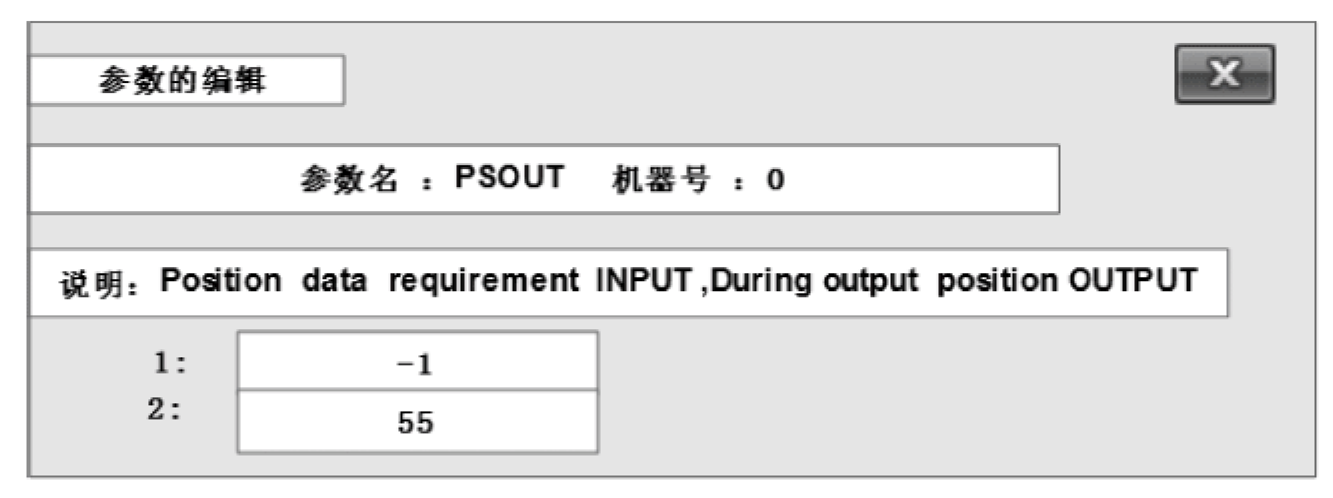
序号	名称	功　能	对应参数	出厂值(端子号)
46	输出的“位置数据编号”	用输出端子“起始位”“结束位”表示“输出位置数据的编号”	PSNUM	

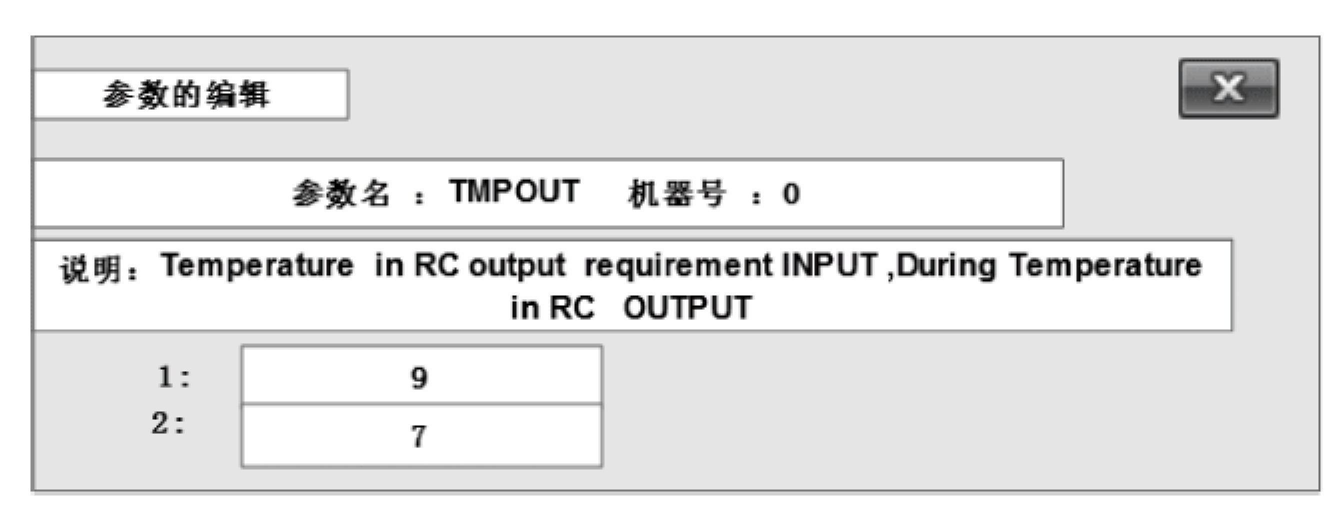

参数的编辑
参数名 ：PSNUM　机器号 ：0
说明：Position number (start,end)INPUT ,position number (start,end)　OUTPUT

1:	30
2:	34
3	40
4	44

图中，设置对应本功能的输出端子号=40～44，输出端子 40～44 构成的数据表示了“输出位置数据的编号”

续表

<table>
<tr><th>序号</th><th>名称</th><th>功　　能</th><th>对应参数</th><th>出厂值(端子号)</th></tr>
<tr><td>47</td><td>“位置数据”的输出状态</td><td>表示当前是否处于“位置数据的输出状态”</td><td>PSOUT</td><td></td></tr>
<tr><td colspan="5">

图中，设置对应本功能的输出端子号＝55，如果机器人当前处于“位置数据的输出状态”，则输出端子55＝ON</td></tr>
<tr><td>48</td><td>控制柜温度输出状态</td><td>表示当前处于“控制柜温度输出状态”</td><td>TMPOUT</td><td></td></tr>
<tr><td colspan="5">

图中，设置对应本功能的输出端子号＝7，如果机器人当前处于“控制柜温度输出状态”，则输出端子7＝ON</td></tr>
</table>

5.4　初步学习 RT ToolBox2 软件

RT ToolBox2 软件(以下简称 RT)是一款专门用于三菱机器人编程、参数设置、程序调试、工作状态监视的软件，其功能强大，编程方便。RT ToolBox2 软件可以在官网上下载。本章先对 RT 软件的使用做一简明的介绍，在本书的第 30 章中将有详细介绍。

设置具体参数的操作方法如下。

单击“离线”→“参数”→“信号参数”→“专用输入输出信号分配”→“通用 1”，弹出如图 5-2 所示的专用输入输出信号设置框。在专用输入输出信号设置框内，可以设置相关的输入输出信号。

这一部分是最常用的参数设置界面。可以在这个窗口中直接设置。这个窗口的设置内容比较集中，也可以如同 5.3.2 节所示，直接打开对应的参数，进行设置。关于参数的学习要在第 16～18 章进行。在这些章节中，对参数进行了专门的解释。



图 5-2　专用输入输出信号设置框

5.5　需要思考的问题

(1) 输入输出端子的序号与功能的关系是固定的吗?

(2) 输入信号 START 与输出信号 START 有什么区别?

(3) 输入信号“操作权”有什么功能?

(4) “程序复位”输入信号有什么功能?

(5) 试用“示教单元”设置输入信号 START。

第6章　第6日——简单的运动指令

【学习目的】

在第3日的学习中，虽然已经让机器人动起来了，但只是一种手动操作，机器人的自动运行才是学习的目的。为了让机器人自动运行，必须学会编写程序。本章的学习目的是学习一些最简单的指令，这些指令是构成自动程序的基础。

6.1　关节插补指令

“关节插补”是机器人运动的一种最常见的方式。机器人在从“起点(当前点)”向“终点”运行时，以各轴等量旋转角度的联动方式实现从“起点(当前点)”到达“终点”，其运行轨迹无法确切描述。“关节插补”是机器人最常用的运动方式。

1. 关节插补的指令格式

```
Mov  <终点>[,<近点>][轨迹类型 Type<常数 1>,Type<常数 2>][<附随语句>]
```

2. 例句

```
Mov (Plt 1,10), 100 Wth M_Out(17) = 1
```

3. 说明

MOV语句是关节插补指令，从“当前点”移动到“终点”。

(1) 终点指“目标点”。

(2)“近点”指接近“终点”的一个点。在实际工作中，往往需要快进到终点的附近，再慢速运动到终点。“近点”在“终点”的Z轴方向位置。根据符号确定是上方或下方。使用近点设置，是一种快速定位的方法。“近点”是“快进”与“慢进”的分界点。

(3)“类型常数Type”用于设置运行轨迹。

①“类型常数Type 1”＝1：绕行。

②“类型常数Type 1”＝0：捷径运行。

“绕行”是指按示教轨迹，可能大于180°轨迹运行。“捷径”指按最短轨迹，即小于180°轨迹运行。

(4) 附随语句。

附随语句如Wth、IFWITH，指在执行MOV指令时，同时执行其他的指令。

4. 样例程序

以下程序中，单引号以后的部分是注释，可做中文注释。

Mov P1'——移动到 P1 点。

Mov P1+P2'——移动到 P1+P2 的位置点。

Mov P1 * P2'——移动到 P1 * P2 位置点(位置点乘法)。

Mov P1,-50'——移动到 P1 点上方 50mm 的“近点”。

Mov P1 Wth M_Out(17)=1'——向 P1 点移动同时指令输出信号(17)=ON。

Mov P1 WthIf M_In(20)=1,Skip'——向 P1 点移动的同时,如果输入信号(20)=ON,就跳到下一行。

Mov P1 Type 1,0' ——指定运行轨迹类型为“捷径型”。

如图 6-1 所示的移动路径其程序如下。

```
Mov P1'——移动到 P1 点。
Mov P2,-50'——移动到 P2 点上方 50mm 位置点即"近点"。
Mov P2'——移动到 P2 点。
Mov P3,-100,Wth M_Out(17)=1'——移动到 P3 点上方 100mm 的"近点"位置,同时指令输出信号(17)=ON。
Mov P3'——移动到 P3 点
Mov P3,-100'——移动到 P3 点上方 100mm 位置点。
End'——程序结束。
```

注意:近点位置以 TOOL 坐标系的 Z 轴方向确定。

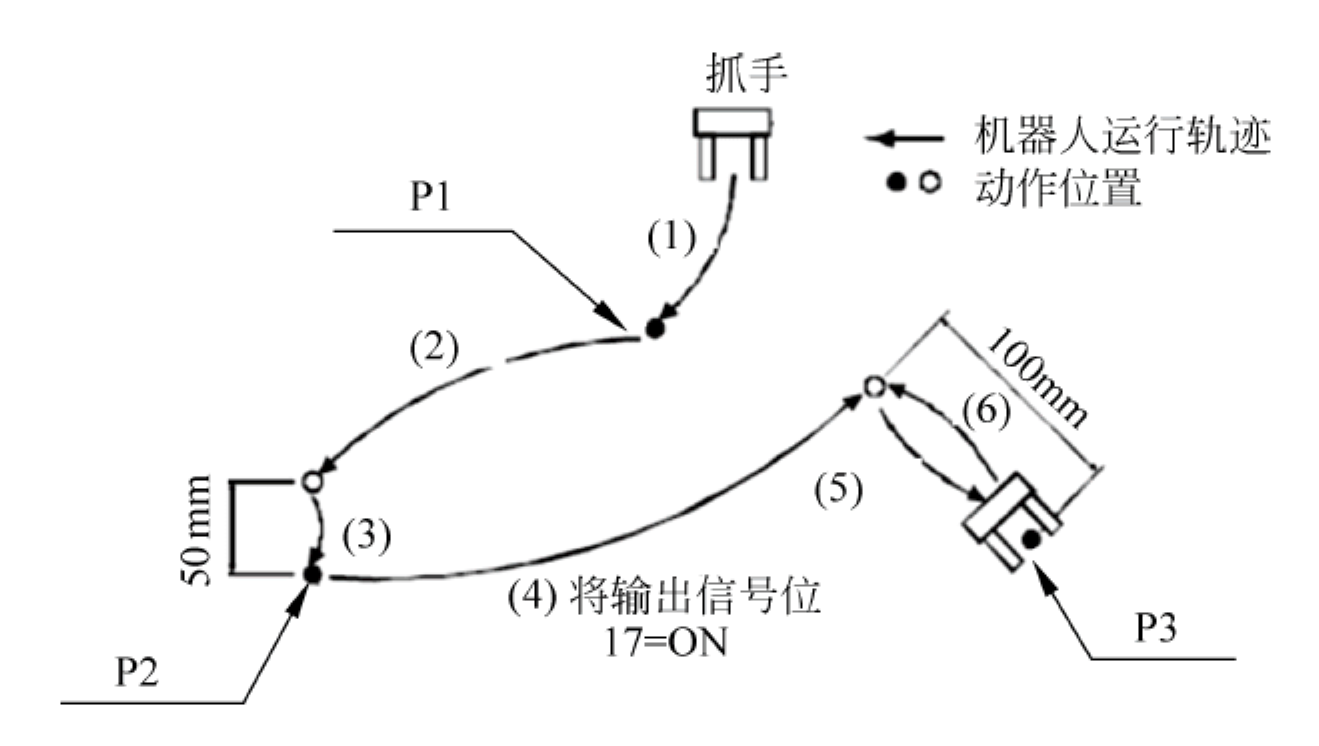


图 6-1　程序及移动路径

6.2 “直线插补”指令

“直线插补”是从“当前点”向“终点”运动的形式。其特点是运行轨迹为直线。这是与关节插补 MOV 指令最大的不同之处。直线插补的运动指令为:Mvs。在需要有明确的直线运动轨迹时,必须使用直线插补指令。

1. 指令格式 1

```
Mvs <终点>,<近点距离>,[<轨迹类型常数 1>,<插补类型常数 2>][<附随语句>]
```

2. 指令格式 2

```
Mvs <离开距离> [<轨迹类型常数 Type1>,<插补类型常数 2>][<附随语句>]
```

注意：这是从“终点”退回“近点”使用的简易指令格式。

3. 对指令格式的说明

(1) <终点>：目标位置点。

(2) <近点距离>：以 TOOL 坐标系的 Z 轴为基准，到“终点”的距离。

(3) <轨迹类型常数 Type1 >，常数 1＝1，绕行；常数 1＝0，捷径运行。

(4) 插补类型：常数＝0，关节插补；常数＝1，直角插补；常数＝2，通过特异点。

(5) <离开距离>：在指令格式 2 中的离开距离是以 TOOL 坐标系的 Z 轴为基准，离开“终点”的距离。这是一个便捷指令。

插补指令的运行轨迹如图 6-2 所示。

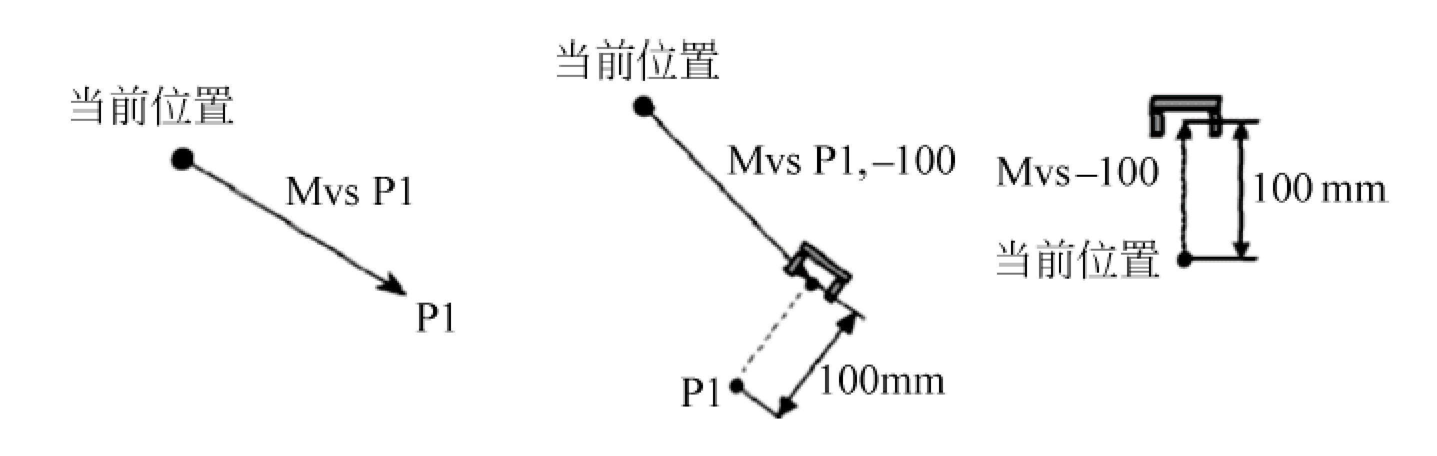


图 6-2　Mvs 指令的移动轨迹

4. 指令例句 1

向终点做直线运动。

```
1  Mvs P1
```

5. 指令例句 2

向“接近点”做直线运动，实际到达“接近点”，同时指令输出信号(17)＝on。

```
1  Mvs P1, - 100.0 Wth M_Out(17) = 1
```

6. 指令例句 3

向终点做直线运动(终点＝P4＋P5，“终点”经过加运算)，实际到达“接近点”，同时如果输入信号(18)＝ON，则指令输出信号(20)＝ON。

```
1  Mvs P4 + P5, 50.0 WthIf M_In(18) = 1, M_Out(20) = 1
```

7. 指令例句 4

从当前点，沿 TOOL 坐标系 Z 轴方向移动 100mm：

```
Mvs , - 100
```

6.3　“真圆插补”指令

“真圆插补”是指机器人的运行轨迹为一真圆。

三维真圆插补指令格式为：Mvc(Move C)。说明如下。

1. 功能

Mvc 指令的运动轨迹是一完整的真圆。需要指定起点和圆弧中的两个点。运动轨迹

如图 6-3 所示。

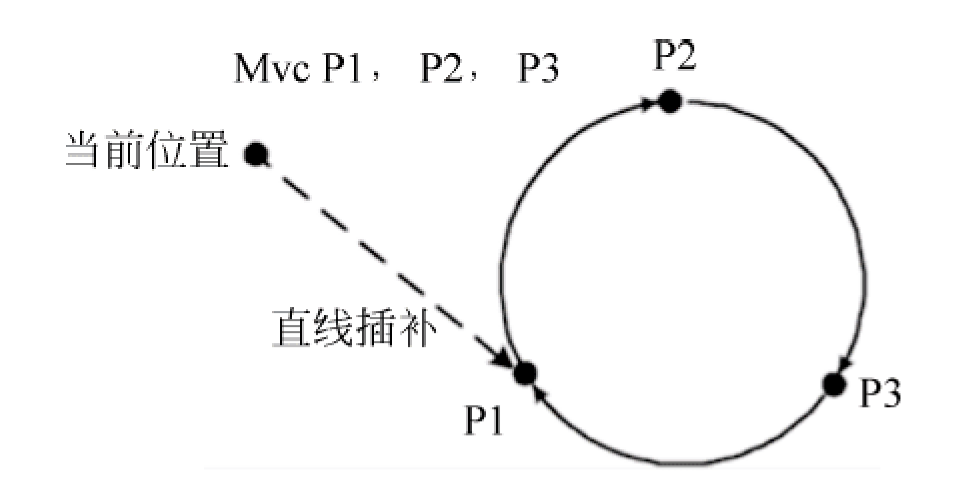


图 6-3 Mvc——三维真圆插补指令的运行轨迹

2. 指令格式

```
Mvc <起点>,<通过点 1>,<通过点 2> 附随语句
```

3. 术语

（1）<起点>,<通过点 1>,<通过点 2>是圆弧上的三个点。

（2）<起点>：真圆的“起点”和“终点”。

4. 运动轨迹

从“当前点”开始到 P1 点，是直线轨迹。真圆运动轨迹为 P1—P2—P3—P1。

5. 指令例句

```
1  Mvc P1,P2,P3'——真圆插补。
2  Mvc P1,J2,P3'——真圆插补。
3  Mvc P1,P2,P3 Wth M_Out(17) = 1'——真圆插补同时输出信号 17 = ON。
4 Mvc P3,(Plt 1,5),P4 WthIf M_In(20) = 1,M_Out(21) = 1'——真圆插补同时如果输入信号 20 = 1,则
输出信号 21 = ON。
```

6. 说明

（1）Mvc 指令的运动轨迹是由指定的三个点构成完整的真圆。

（2）圆弧插补的“形位”为起点“形位”。通过其余两点的“形位”不计。

（3）从“当前点”开始到 P1 点，是直线插补轨迹。

6.4 启动和停止信号

在将以上的指令输入机器人控制器后，如何发出“启动信号”和“停止信号”呢？这必须给输入输出端子赋予功能，没有赋予“功能”之前，即使输入输出端子已经接线完毕，仍然是“空白”的。对输入输出端子赋予功能的方法是使用 RT 软件进行设置，操作方法如下。

（1）将计算机连接到“机器人控制器，打开 RT 软件”。

（2）单击“离线”→“参数”→“信号参数”→“专用输入输出信号分配”→“通用 1”，弹出如图 6-4 所示的专用输入输出信号设置框，在专用输入输出信号设置框内，可以设置相关的输入输出信号。

（3）如图 6-5 所示，将输入信号端子 3 设置为“启动”；将输入信号端子 0 设置为“停止”。将这两个输入信号分别与操作面板上的按钮开关相连接。

（4）在安全状态下，按下“启动”按钮和“停止”按钮，观察运动状态。



输入信号(I)			输出信号(U)		
可自动运行	AUTOENA		可自动运行	AUTOENA	
启动	START	3	运行中	START	0
停止	STOP	0	待机中	STOP	
停止(STOP2)	STOP2		待机中2	STOP2	
			停止输入中	STOPSTS	
程序复位	SLOTINIT		可以选择程序	SLOTINIT	
报错复位	ERRRESET	2	报警发生中	ERRRESET	2
周期停止	CYCLE		周期停止中	CYCLE	
伺服OFF	SRVOFF	1	伺服ON不可	SRVOFF	
伺服ON	SRVON	4	伺服ON中	SRVON	1
操作权	IOENA	5	操作权	IOENA	3

图 6-4 设置启动和停止信号

输入信号(I)			输出信号(U)		
可自动运行	AUTOENA		可自动运行	AUTOENA	
启动	START	3	运行中	START	0
停止	STOP	0	待机中	STOP	
停止(STOP2)	STOP2		待机中2	STOP2	
			停止输入中	STOPSTS	
程序复位	SLOTINIT		可以选择程序	SLOTINIT	
报错复位	ERRRESET	2	报警发生中	ERRRESET	2
周期停止	CYCLE		周期停止中	CYCLE	
伺服OFF	SRVOFF	1	伺服ON不可	SRVOFF	
伺服ON	SRVON	4	伺服ON中	SRVON	1
操作权	IOENA	5	操作权	IOENA	3

图 6-5 设置输入输出信号

6.5 实 验

(1) 编制一段运动程序,如图 6-6 所示。

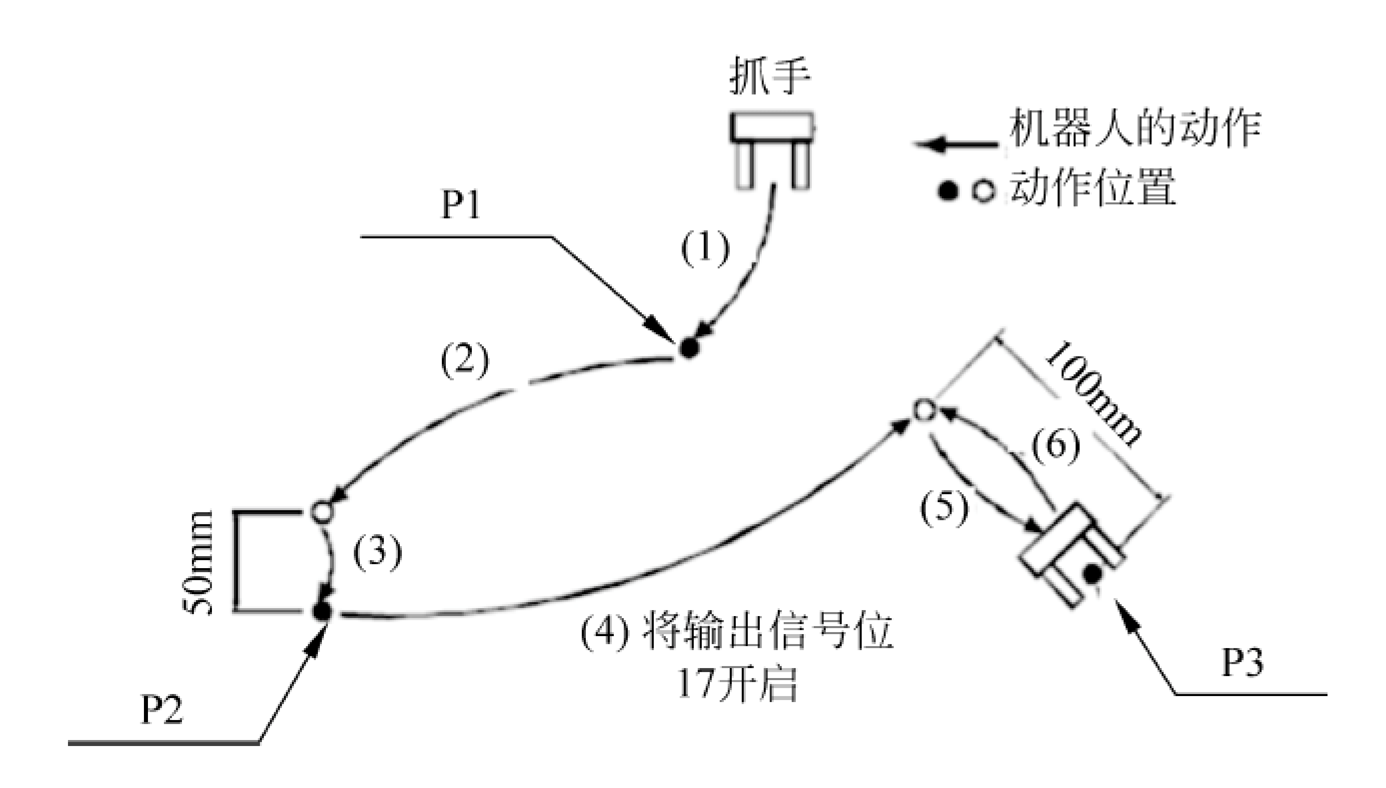


图 6-6 程序及移动路径

(2) 参照 4.4 节,示教确定 P1、P2、P3 点。

(3) 设置输入信号“启动”和“停止”。

(4) 在安全状态下(将“急停”按钮置于控制范围),将速度倍率设置=10,发出“启动”和

“停止”信号，观察运动轨迹。

(5) 重新设置输入信号“启动”和“停止”。例如，设置“启动”＝4，“停止”＝9，观察机器人运动状态。

如图6-1所示的移动路径其程序如下。

```
Mov P1'—— 移动到 P1 点。
Mov P2, - 50'—— 移动到 P2 点上方 50mm 位置点即"近点"。
Mov P2'—— 移动到 P2 点。
Mov P3, - 100,Wth M_Out(17) = 1'—— 移动到 P3 点上方 100mm 的"近点"位置,同时指令输出信号(17) = ON。
Mov P3'—— 移动到 P3 点。
Mov P3, - 100'—— 移动到 P3 点上方 100mm 位置点。
End'——程序结束。
```

6.6 需要思考的问题

(1) 什么是插补运行？

(2) “MOV 指令”与“Mvs 指令”都是点对点的运动指令，这两个指令有什么区别？

(3) 什么是“近点”？为什么通常要使用“近点”？

(4) 如何确定各个点的位置？

(5) 如何设置输入输出信号的功能？

第 7 章　第 7 日——机器人的坐标系

【学习目的】

在第 6 章的学习中，通过简短的程序已经使机器人运动起来，但是各个位置点的基准在哪里呢？换句话说，机器人是按照什么样的坐标系运动的呢？在一般的运动机械中，都是按照直角坐标系运动。直角坐标系的原点由机床上的某个基准点确定。机器人也是运动机械，机器人的坐标系如何确定呢？本章就是要学习机器人的坐标系。由于机器人结构的特殊性，机器人所使用的坐标系比一般工作机械要复杂些，但是使用不同的坐标系本身还是要使机器人的运动编程更简单。

7.1　基本坐标系

基本坐标系是以机器人底座安装基面为基准的坐标系。在机器人底座上有图示标志。基本坐标系如图 7-1 所示。实际上，“基本坐标系”是机器人的第一基准坐标系，“世界坐标系”也是以“基本坐标系”为基准的。

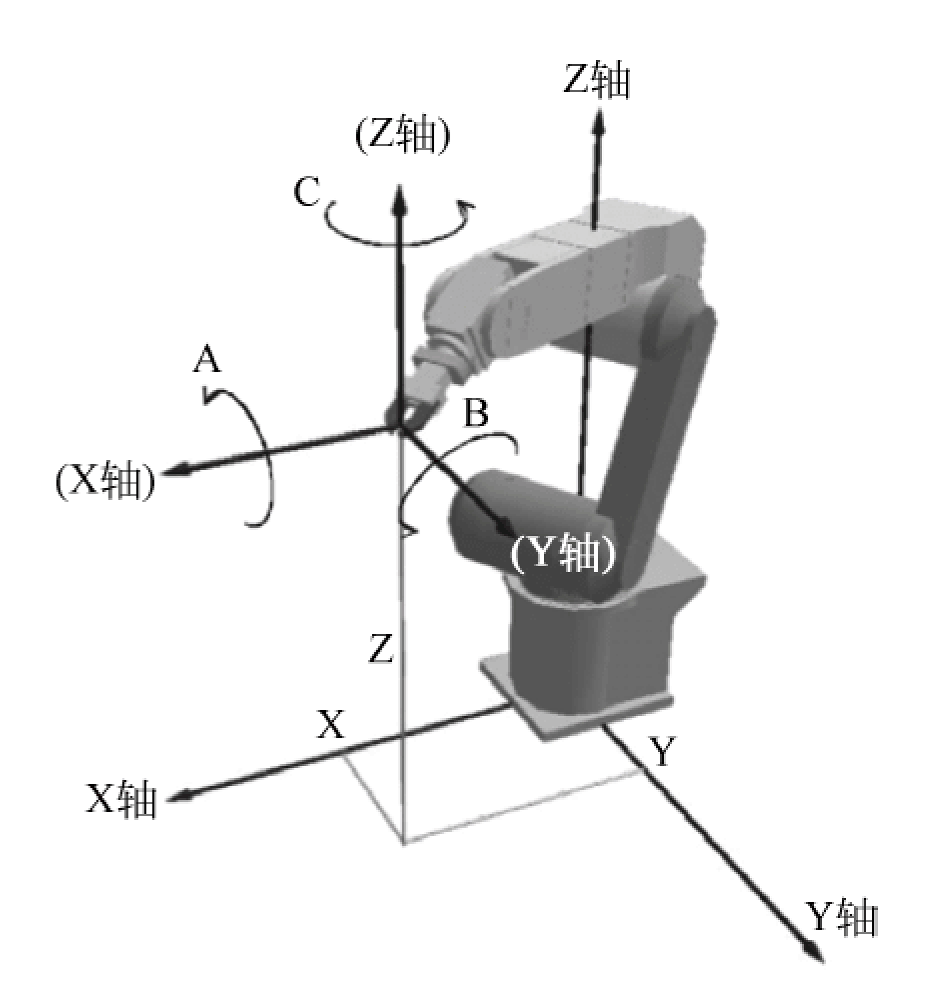


图 7-1　基本坐标系

当机器人的安装位置确定以后，“基本坐标系”就确定了。“基本坐标系”是机器人诸多坐标系的基准。

7.2 世界坐标系

"世界坐标系"是机器人系统默认使用的坐标系，是表示机器人(控制点)位置的"当前坐标系"。所有表示位置点的数据都是以"世界坐标系"为基准的。("世界坐标系"类似于数控系统的 G54 坐标系，事实上就是"工件坐标系"。)

"世界坐标系"是以机器人的"基本坐标系"为基准设置的(这是因为每一台机器人的"基本坐标系"是由其安装位置决定的)。只是确定"世界坐标系"基准点时，是从"世界坐标系"来观察"基本坐标系"的位置，从而确定新的"世界坐标系"本身的基准点，所以"基本坐标系"是机器人坐标系中第 1 基准坐标系。

在大部分的应用中，"世界坐标系"与"基本坐标系"相同。

如图 7-2 所示，图中 $X_W Y_W Z_W$ 是"世界坐标系"。当前位置是以世界坐标系为基准的，如图 7-3 所示。

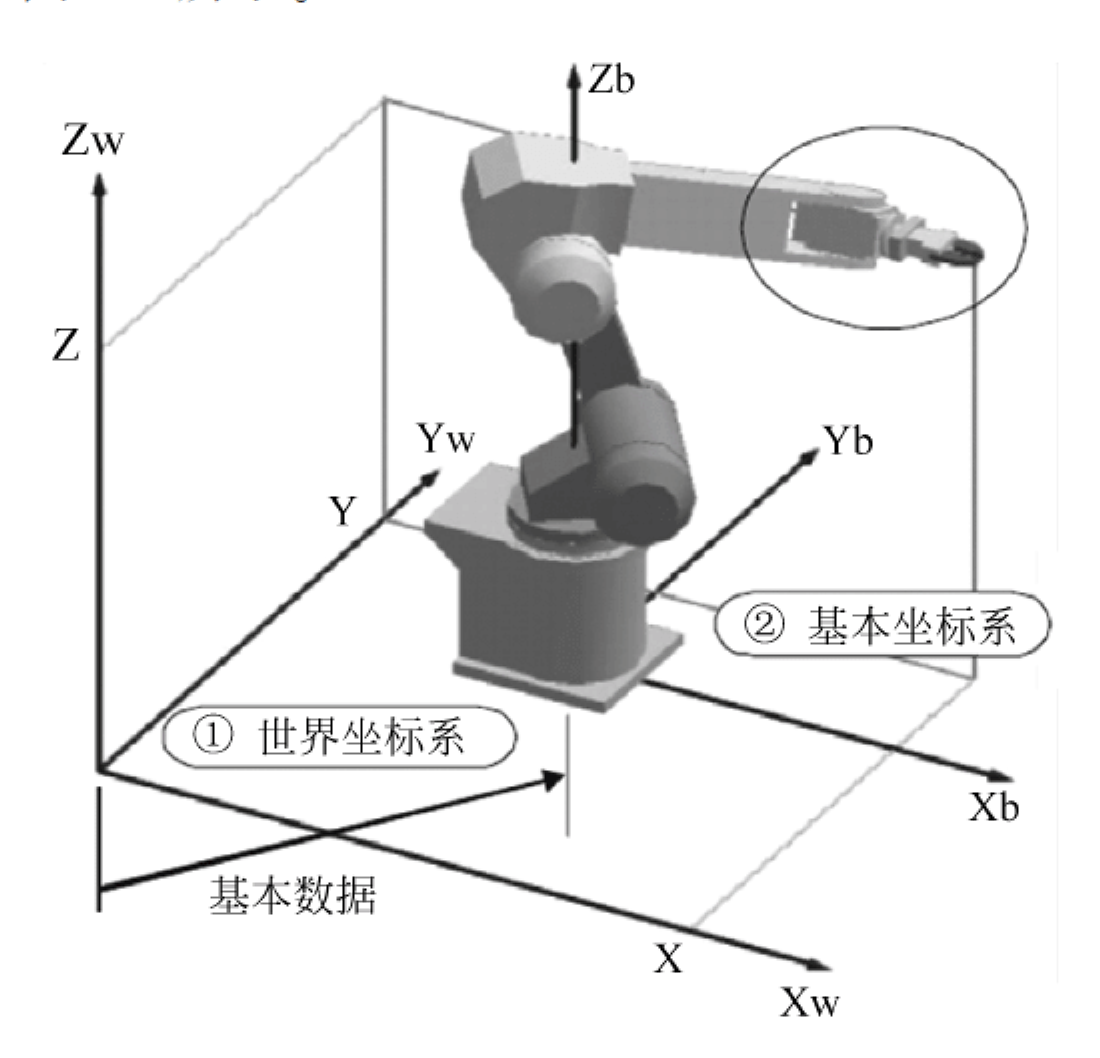


图 7-2 "世界坐标系"与"基本坐标系"之间的关系

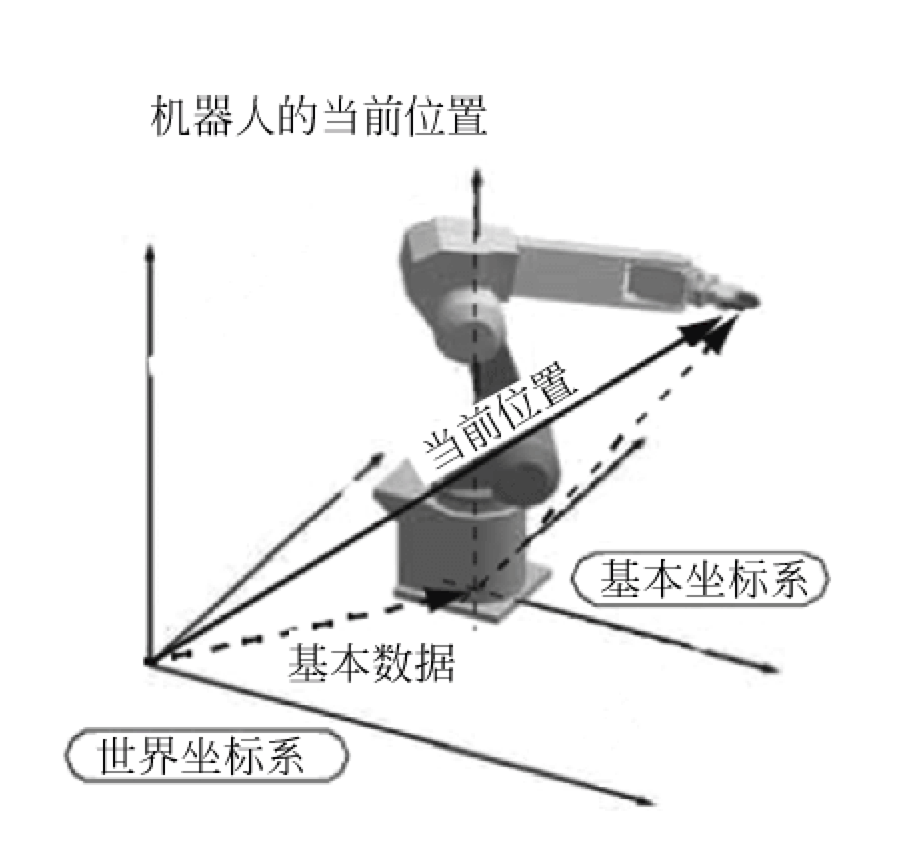


图 7-3 "当前位置"以"世界坐标系"为基准

7.3 机械 IF 坐标系

"机械 IF 坐标系"也就是"机械法兰面坐标系"。以机器人最前端法兰面为基准确定的坐标系称为"机械 IF 坐标系"，以 Xm-Ym-Zm 表示，如图 7-4 所示。与法兰面垂直的轴为"Z 轴"，Z 轴正向朝外，Xm 轴、Ym 轴在法兰面上。法兰中心与定位销孔的连接线为 Xm 轴，但必须注意 Xm 轴的"正向"与定位销孔相反。

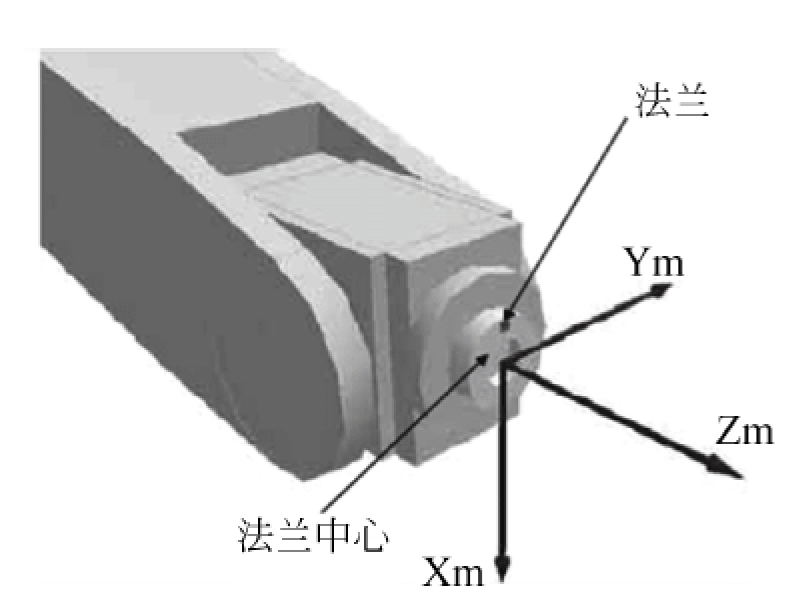


图 7-4 机械 IF 坐标系的定义

由于在机械法兰面要安装抓手，所以这个"机械法兰面"就具有特殊意义。特别注意：机械法兰面转动，机械 IF 坐标系也随之转动。而法兰面的转动受 J4 和

J6 轴的影响(特别是 J6 轴的旋转带动了法兰面的旋转,也就带动了机械 IF 坐标系的旋转,如果以机械 IF 坐标系为基准执行定位,就会影响很大),参见图 7-5 和图 7-6。图 7-6 是 J6 轴逆时针旋转了的坐标系。

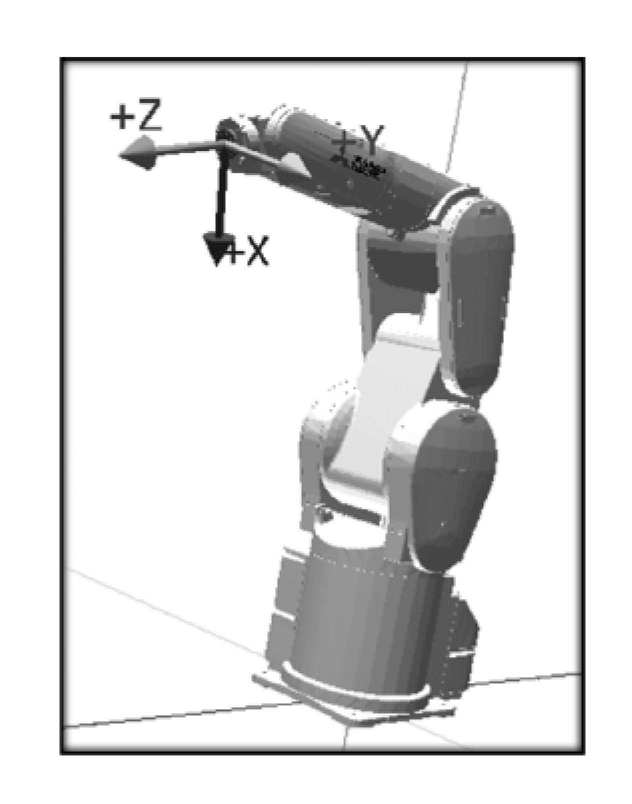


图 7-5 机械 IF 坐标系的图示

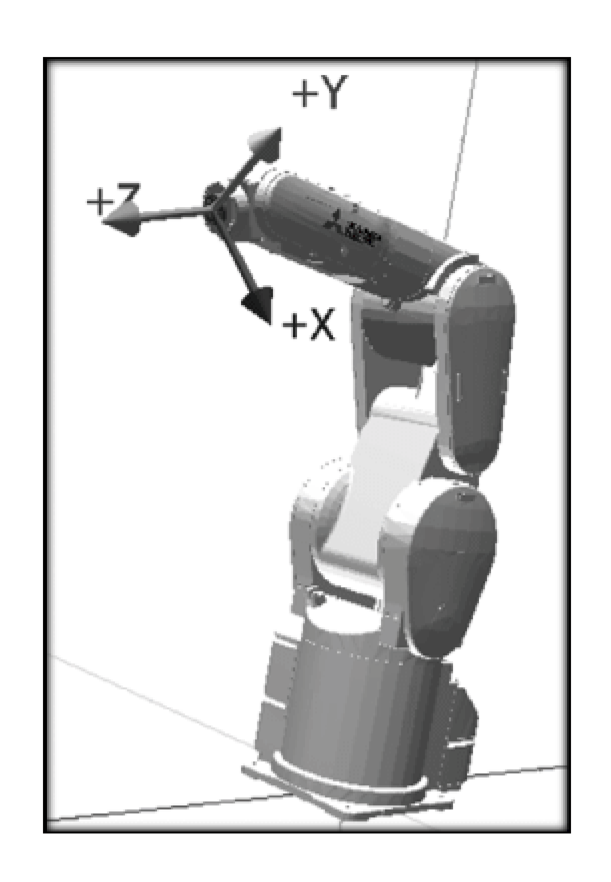


图 7-6 J6 轴逆时针旋转了的机械 IF 坐标系

7.4 TOOL 坐标系

7.4.1 定义及设置

工具(TOOL)坐标系的定义及设置基准如下。

1. 定义

由于实际使用的机器人都要安装夹具抓手等辅助工具,所以,机器人的实际控制点就移动到了工具的中心点上,为了控制方便,以工具的中心点为基准建立的坐标系就是 TOOL 坐标系。

2. 设置

由于夹具抓手直接安装在机械法兰面上,所以 TOOL 坐标系就是以机械 IF 坐标系为基准建立的。建立 TOOL 坐标系有参数设置方法和指令设置法,实际上都是确定 TOOL 坐标系原点在机械 IF 坐标系中的位置和形位。

TOOL 坐标系与机械 IF 坐标系的关系如图 7-7 所示。TOOL 坐标系用 Xt、Yt、Zt 表

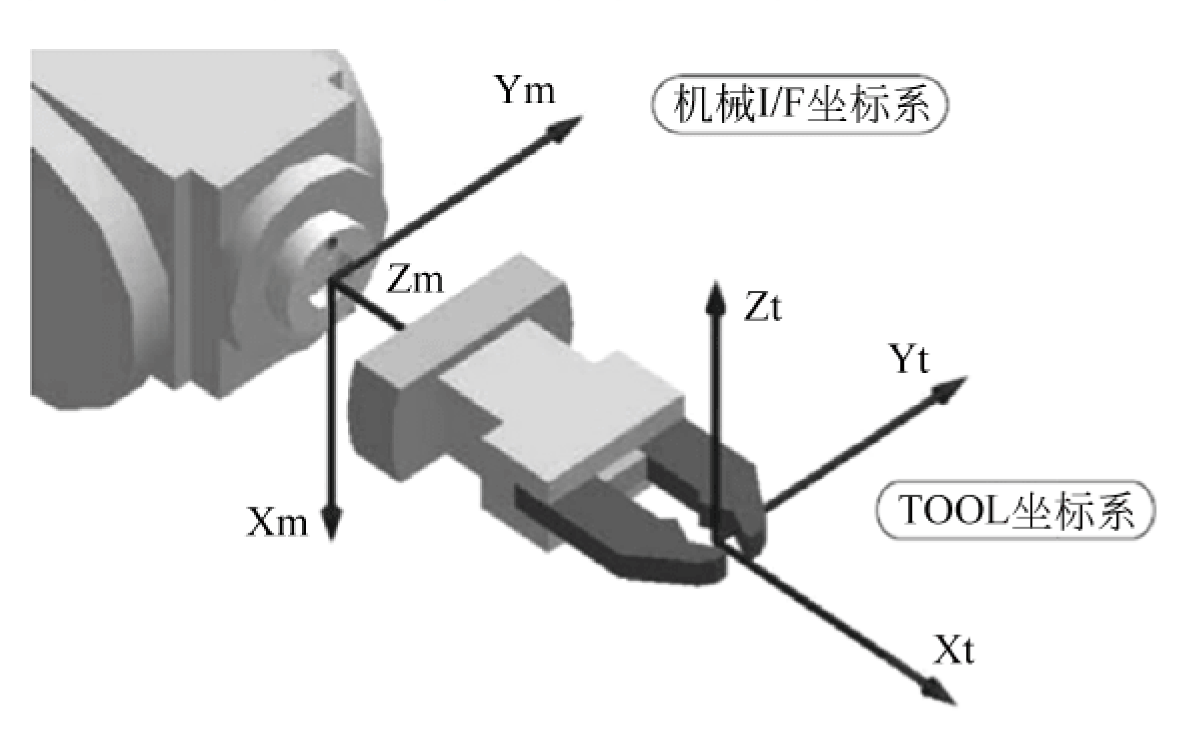


图 7-7 工具坐标系与机械 IF 坐标系的关系

示。TOOL 坐标系是在机械 IF 坐标系基础上建立的。在 TOOL 坐标系的原点数据中，XYZ 表示 TOOL 坐标系原点在机械 IF 坐标系内的直交位置点。ABC 表示 TOOL 坐标系绕机械 IF 坐标系 Xm、Ym、Zm 轴的旋转角度。

TOOL 坐标系的原点不仅可以设置在“任何”位置，而且坐标系的形位也可以通过 ABC 值任意设置(相当于一个立方体在一个万向轴接点任意旋转)。在图 7-7 中，TOOL 坐标系绕 Y 轴旋转了 −90°，所以 Zt 轴方向就朝上(与机械 IF 坐标系中的 Zm 方向不同)。而且当机械法兰面旋转(J6 轴旋转)时，TOOL 坐标系也会随着旋转，分析时要特别注意。

7.4.2 动作比较

1. JOG 或示教动作

1) 使用机械 IF 坐标系

未设置 TOOL 坐标系时，使用机械 IF 坐标系以出厂值法兰面的中心为“控制点”，在 X 方向移动(此时，X 轴垂直向下)，其移动形位如图 7-8 所示。

2) 以 TOOL 坐标系动作

设置了 TOOL 坐标系后，以 TOOL 坐标系动作。注意在 X 方向移动时，是沿着 TOOL 坐标系的 Xt 方向动作，这样就可以平行或垂直于抓手面动作，使 JOG 动作更简单易行，如图 7-9 所示。

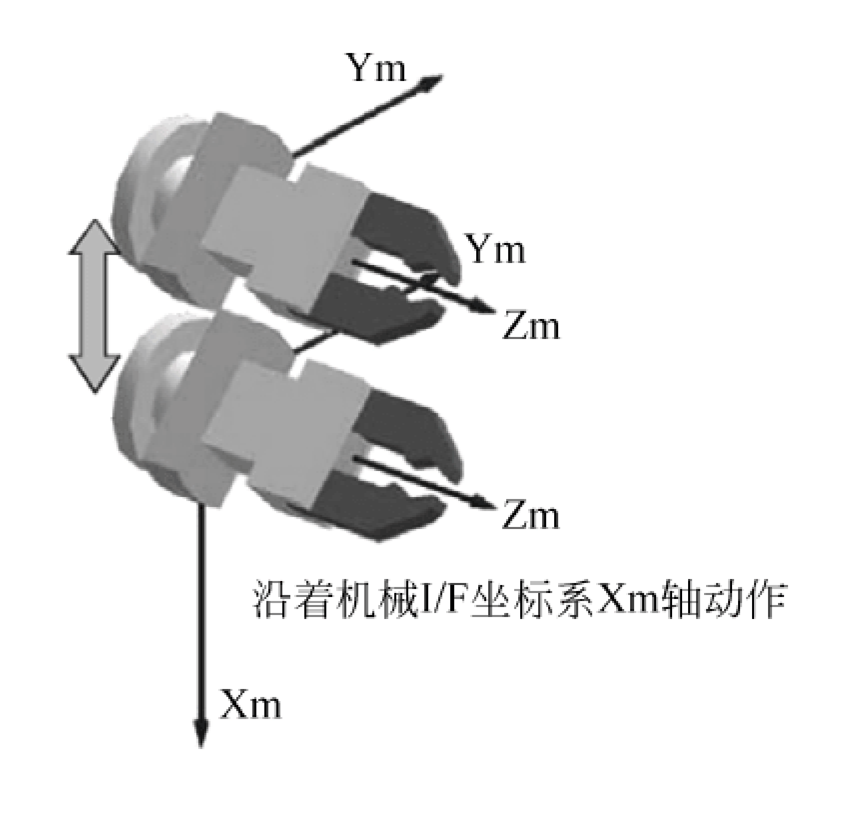


图 7-8　X 方向移动的形位

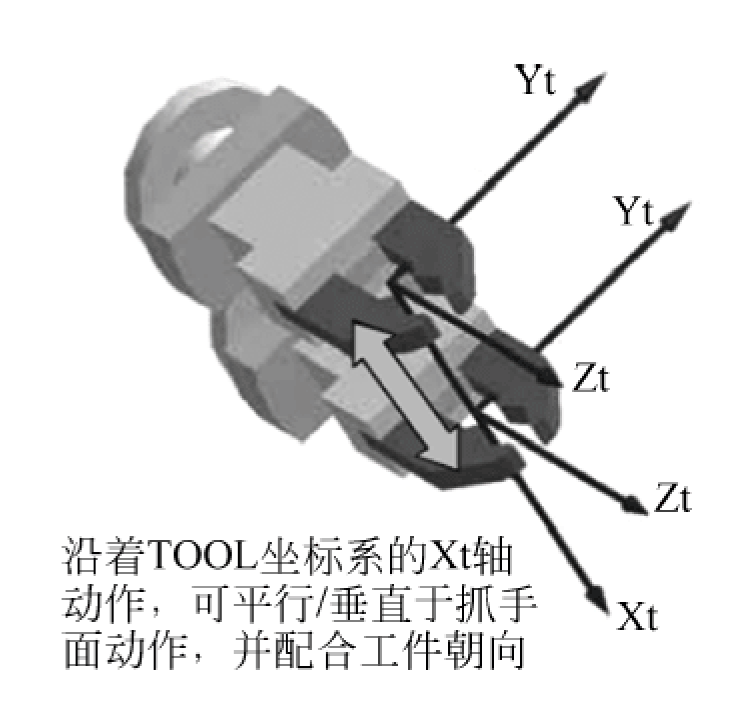


图 7-9　在 TOOL 坐标系 X 方向移动

3) A 方向动作

(1) 使用机械 IF 坐标系

未设置 TOOL 坐标系时，使用机械 IF 坐标系，绕 Xm 轴旋转，抓手前端大幅度摆动，如图 7-10 所示。

(2) 设置 TOOL 坐标系绕 Xt 轴旋转

设置 TOOL 坐标系后，绕 Xt 轴旋转，抓手前端绕工件旋转，在不偏离工件位置的情况下，改变机器人形位，如图 7-11 所示。

以上是在 JOG 运行时的情况。

A方向移动

Ym
Ym
Zm
Zm
Xm
机械I/F坐标系的Xm轴旋转，抓手前端大幅移动

图 7-10　A 方向的动作

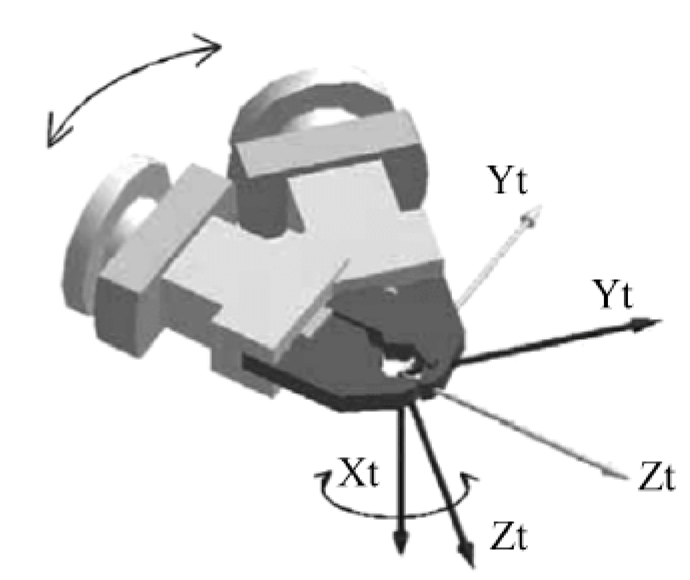


图 7-11　在 TOOL 坐标系中绕 Xt 轴旋转

2. 自动运行

1）近点运行

在自动程序运行时，TOOL 坐标系的原点为机器人“控制点”。在程序中发出的定位点是以世界坐标系为基准的。但是，Mov 指令中的近点运行功能中的“近点”的位置则是以 TOOL 坐标系的 Z 轴正负方向为基准移动。这是必须注意的。

指令样例：

```
Mov P1,50
```

其动作是：将 TOOL 坐标系原点移动到 P1 点的“近点”，“近点”为 P1 点沿 TOOL 坐标系的 Z 轴＋向移动 50mm，如图 7-12 所示。

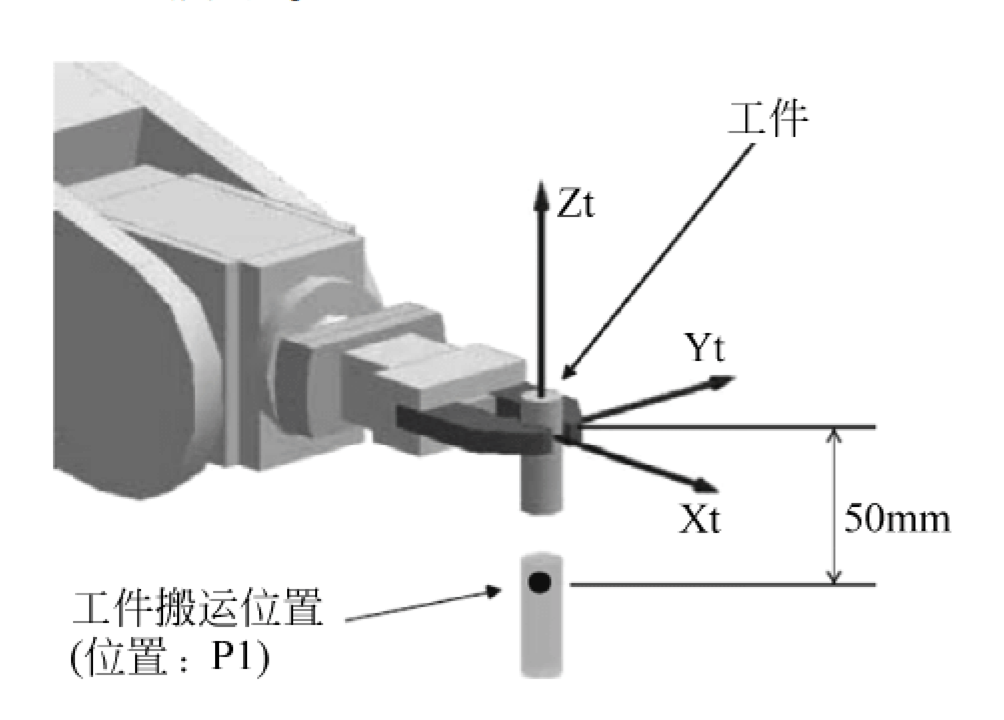


图 7-12　在 TOOL 坐标系中的近点动作

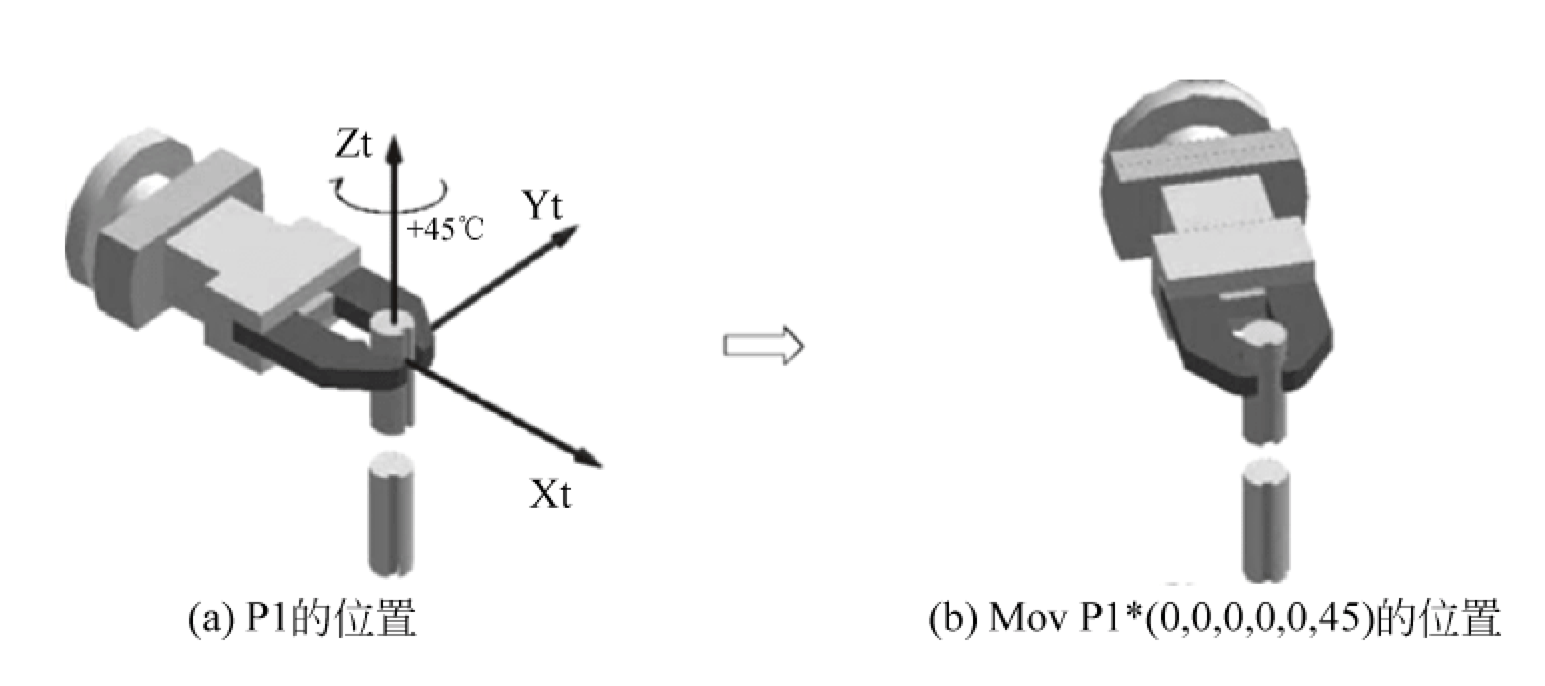
2）相位旋转

绕工件位置点旋转（Zt），可以使工件旋转一个角度。

例如，指令在 P1 点绕 Z 轴旋转 45°（使用两点的乘法指令）。

```
Mov P1 * (0,0,0,0,0,45)'——注意：使用两点的乘法指令
```

实际的运动结果如图 7-13 所示。



(a) P1的位置　　(b) Mov P1*(0,0,0,0,0,45)的位置

图 7-13　在 TOOL 坐标系中的相位旋转

7.5　工件坐标系

工件坐标系是以工件原点确定的坐标系。在实际加工中,工件的图纸是绘制完毕的。如果要走出工件的轨迹,当然以工件尺寸直接编程最为简捷。这就需要一个以工件原点为基准的坐标系,这就是工件坐标系。

在机器人系统中,可以通过参数预先设置 8 个工件坐标系。

1. 参数设置法

表 7-1 为工件坐标系相关参数,可在软件上做具体设置。

表 7-1　工件坐标系相关参数

类　　型	参数符号	参数名称	功　　能
动作	WKnCORD n=1～8	工件坐标系	设置工件坐标系
	WKnWO	工件坐标系原点	
	WKnWX	工件坐标系 X 轴位置点	
	WKnWY	工件坐标系 Y 轴位置点	
设置	可设置 8 个工件坐标系		



2. 指令设置法

设置世界坐标系的偏置坐标(偏置坐标为以世界坐标系为基准观察到的基本坐标系原点在世界坐标系内的坐标)。

样例程序如下。

```
1  Base (50,100,0,0,0,90)'——设置一个新的世界坐标系(如图 7-14 所示)。
2  Mvs P1'——前进到 P1 点。
3  Base P2'——以 P2 点为偏置量,设置一个新的世界坐标系。
4  Mvs P1  '——前进到 P1 点。
5  Base 0  设置世界坐标系与基本坐标系相同(回初始状态)。
```

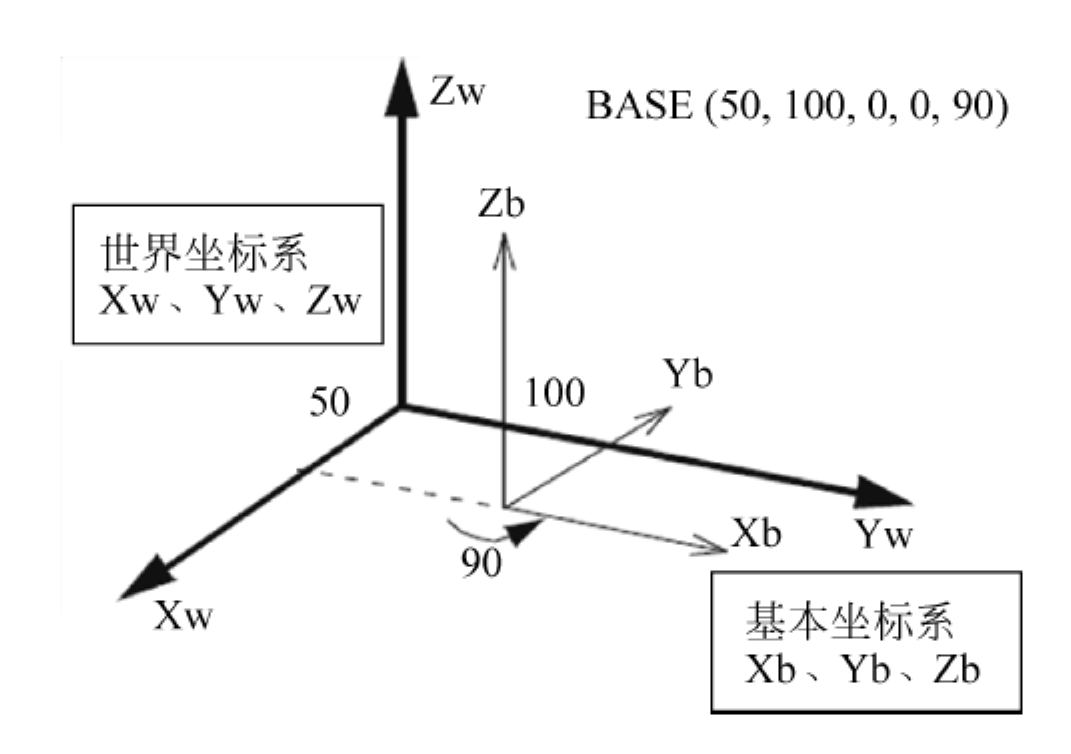


图 7-14　使用 Base 指令设置新的坐标系

3. 以工件坐标系号选择新世界坐标系的方法

样例程序如下。

```
1  Base 1'——选择 1# 工件坐标系 WK1CORD。
2  Mvs P1'——运动到 P1。
3  Base 2'——选择 2# 工件坐标系 WK2CORD。
4  Mvs P1'——运动到 P1。
5  Base 0'——选择基本坐标系。
```

7.6　需要思考的问题

(1) 基本坐标系如何确定?

(2) 世界坐标系是机器人默认使用的坐标系吗? 世界坐标系与基本坐标系有什么关系?

(3) 什么是坐标系偏置?

(4) 什么是机械 IF 坐标系? 如何确定机械 IF 坐标系的原点?

(5) TOOL 坐标系又称为抓手坐标系吗? 如何设置 TOOL 坐标系?

第 8 章 第 8 日——对机器人系统的初步设置

【学习目的】

本章要学习比较实用的操作。实际上任何一台工作机械在运行之前,必须选择坐标系,设置原点,特别是要设置行程范围,也就是行程限位,以保证工作机械在正常的行程范围内工作,不发生事故。机器人由于其特殊性有多种原点设置和行程范围设置的方法。这就是本章要学习的内容。

8.1 坐标系的选择

在第 7 章中介绍了机器人使用的各种坐标系。而世界坐标系是机器人系统默认使用的坐标系,因此如果不在程序中特别指定,就是使用世界坐标系。如果不特别加以设置,世界坐标系与基本坐标系相同。

8.2 原点的设置

8.2.1 设置原点的方法种类

在开机后,首先必须设置原点。三菱机器人有以下 6 种设置原点方式,如图 8-1 所示。

(1) 原点数据输入方式;

(2) 机械限位器方式;

(3) 夹具方式;

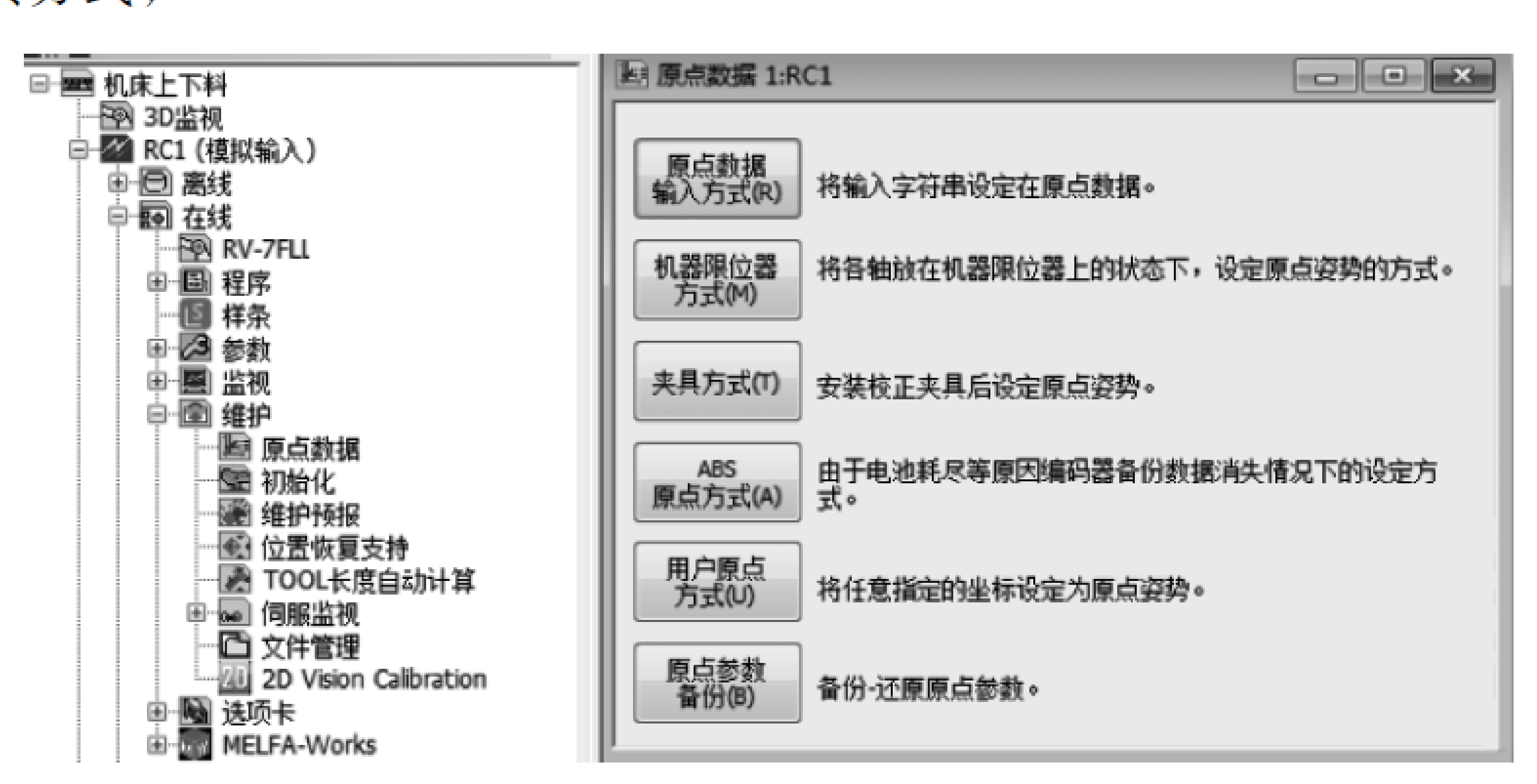


图 8-1 原点数据设置框

（4）ABS原点方式；
（5）用户原点方式；
（6）原点参数备份方式。
使用RT软件可以设置原点(参见第30章)。
单击“维护”→“原点数据”，弹出如图8-1所示原点数据设置框。

8.2.2 原点数据输入方式的使用

原点数据输入方式——直接输入“字符串”，这是最常用的方法。

出厂时，厂家已经标定了各轴的原点，并且作为随机文件提供给使用者。一方面使用者在使用前应该输入“原点文件”——原点文件中每一轴的原点是一个“字符串”，使用者应该妥善保存“原点文件”。另一方面，如果原点数据丢失后，可以直接输入原点文件的字符串，以恢复原点。

本操作需要在联机状态下操作。

（1）打开RT软件，单击“维护”→“原点数据”，弹出如图8-1所示原点数据设置框。
（2）单击“原点数据输入方式”按钮，弹出如图8-2所示“原点数据的设定”对话框。



图8-2 原点数据输入方式——直接输入字符串

（3）根据出厂文件设置，输入“原点数据”。
（4）写入——将设置完毕的数据写入控制器。
（5）保存文件——将当前原点数据保存到计算机中。
（6）从文件读出——从计算机中读出“原点数据文件”。
（7）更新——从控制器内读出“原点数据”，显示最新的原点数据。

8.3 机器人初始化的基本操作

（1）功能：将机器人控制器中的数据进行初始化。
可对下列信息进行初始化。
① 时间设定；

② 删除所有程序；

③ 电池剩余时间的初始化；

④ 控制器的序列号的确认设定。

(2) 操作方法如图 8-3 所示。

① 打开 RT 软件，单击“维护”→“初始化”，弹出如图 8-3 所示“初始化”设置框。

② 对程序进行“初始化”——删除控制器内所有程序。

③ 设定“当前时间”。



图 8-3 初始化操作框

8.4 行程范围设置

操作方法如下。

单击“离线”→“参数”→“动作参数”→“动作范围”，弹出如图 8-4 所示的动作范围设置框。在动作范围设置框内，可以设置各轴的关节动作范围、在直角坐标系内的动作范围等内容，既明确又快捷方便。

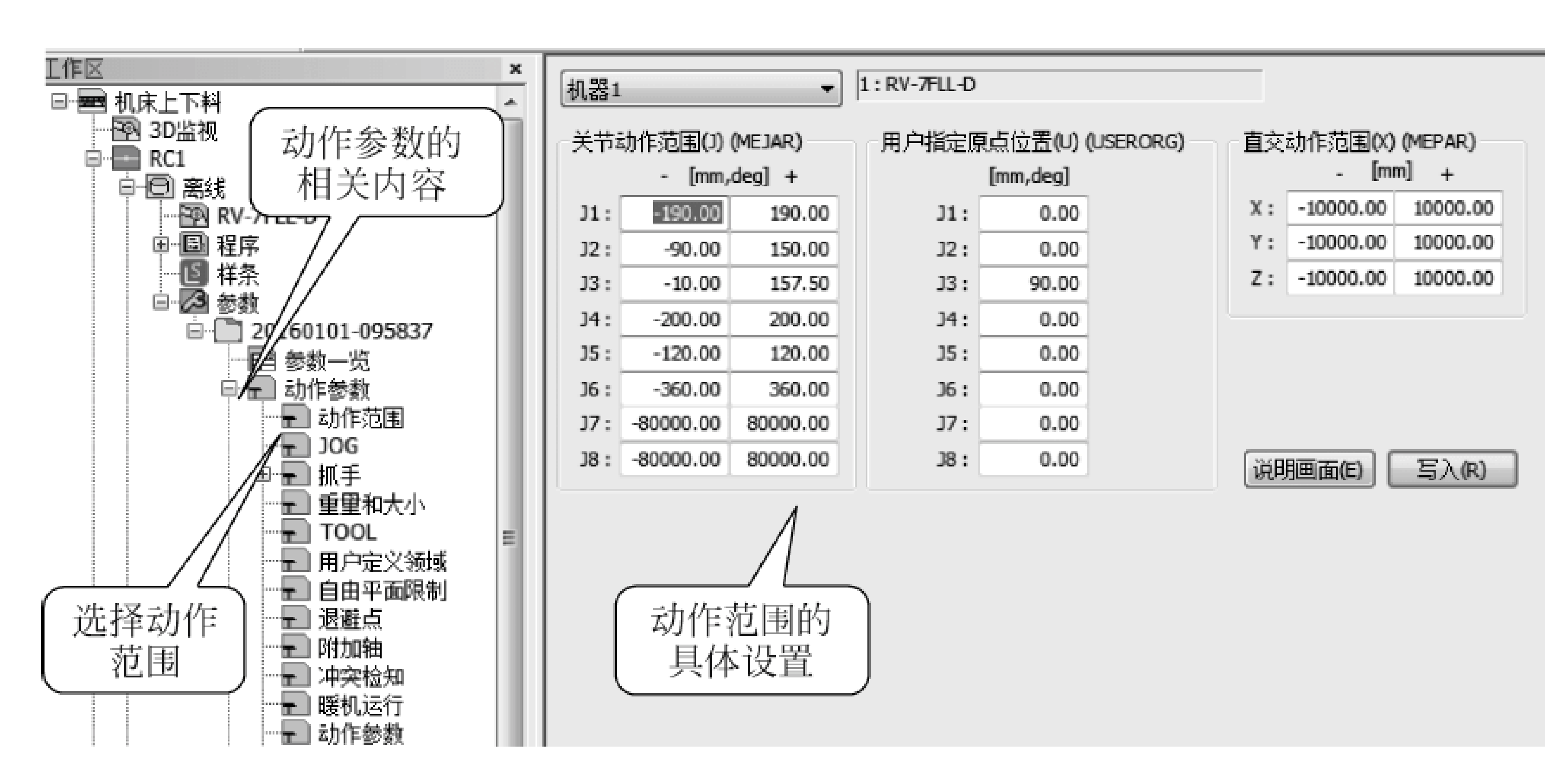


图 8-4 设置具体参数

设置完毕后，单击“写入”按钮，操作完成。

8.5 需要思考的问题

(1) 机器人默认的坐标系是什么坐标系？

(2) 如何设置原点？

(3) 如何进行初始化操作？

(4) 如何用角度设置动作范围？

第 9 章　第 9 日——编程指令的学习和使用(1)

【学习目的】

经过前面 8 章的学习,读者应该已经能够初步使用机器人了。本章以后要学习更多更深的机器人编程指令、状态变量、参数的定义及设置等内容,进入中级学习阶段。

9.1　三维圆弧插补指令 Mvr

1. 功能

Mvr 指令为三维圆弧插补指令,需要指定“起点”和圆弧中的“通过点”和“终点”,运动轨迹是一段圆弧,如图 9-1 所示。

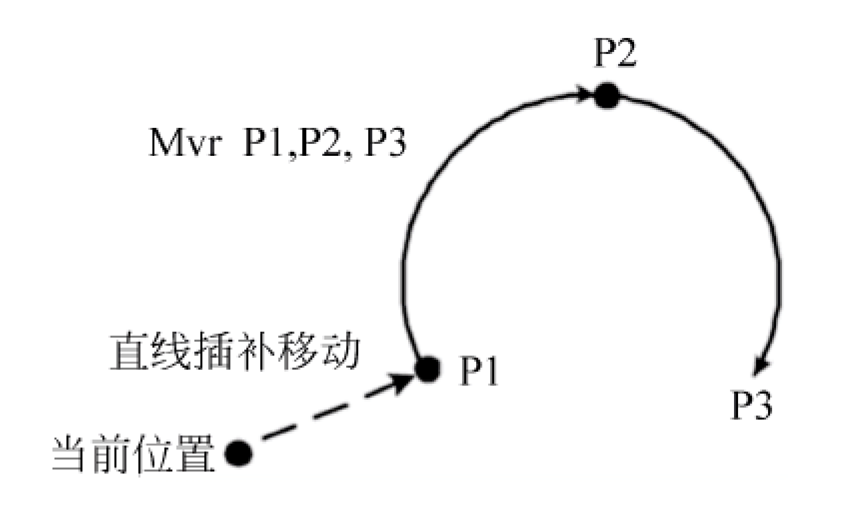


图 9-1　Mvr 指令的运动轨迹

2. 指令格式

```
Mvr  <起点>,<通过点>,<终点><轨迹类型 1 >,<插补类型>附随语句
```

(1) <起点>:圆弧的起点。

(2) <通过点>:圆弧中的一个点。

(3) <终点>:圆弧的终点。

(4) <轨迹类型 1 >:规定运行轨迹是“捷径”还是“绕行”。捷径=0,绕行=1。

(5) <插补类型>:规定“关节插补”或“三轴直交插补”或“通过特异点”。关节插补=0,三轴直交插补=1,通过特异点=2。

3. 指令例句

1　Mvr P1,J2,P3'——圆弧插补。

2　Mvr P1,P2,P3 Wth M_Out(17) = 1'——圆弧插补,同时指令输出信号 17 = ON。

3　Mvr P3,(Plt 1,5),P4 WthIf M_In(20) = 1,M_Out(21) = 1'——圆弧插补,同时如果输入信号 20 = 1,则输出信号 21 = ON。

9.2 “两点型圆弧插补”指令

1. 功能

Mvr2 指令是两点型圆弧插补指令，需要指定“起点”和“终点”以及“参考点”，运动轨迹是一段只通过起点和终点的圆弧，不实际通过参考点（参考点的作用只用于构成圆弧轨迹），如图 9-2 所示。

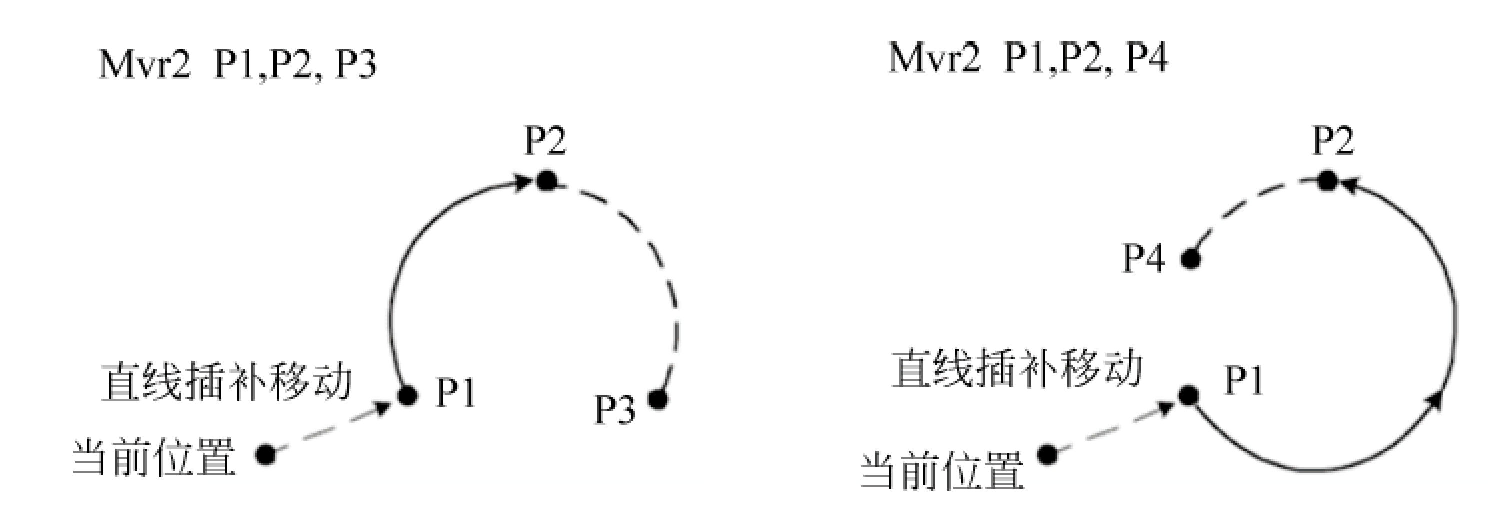


图 9-2 Mvr2 指令的运动轨迹

2. 指令格式

Mvr2 <起点>，，<终点>，<参考点> 轨迹类型，插补类型 附随语句

3. 说明

（1）轨迹类型：常数 1＝1，绕行；常数 1＝0，捷径运行。

（2）插补类型：常数＝0，关节插补；常数＝1，直交插补；常数＝2，通过特异点。

4. 指令例句

```
1  Mvr2 P1,P2,P3'——以 P1,P2,P3 点做圆弧插补。
2  Mvr2 P1,J2,P3'——以 P1,J2,P3 点做圆弧插补。
3  Mvr2 P1,P2,P3 Wth M_Out(17) = 1'——以 P1,J2,P3 点做圆弧插补,同时指令输出信号 17 = ON。
4  Mvr2 P3,(Plt 1,5),P4 WthIf M_In(20) = 1,M_Out(21) = 1'——以 P3, (Plt 1,5),P4 点做圆弧插补,同时如果输入信号 20 = ON,则指令输出信号 21 = ON。
```

9.3 “三点型圆弧插补”指令

1. 功能

Mvr3 指令是三点圆弧插补指令，需要指定起点和终点及圆心点，运动轨迹是一段只通过起点和终点的圆弧，如图 9-3 所示。

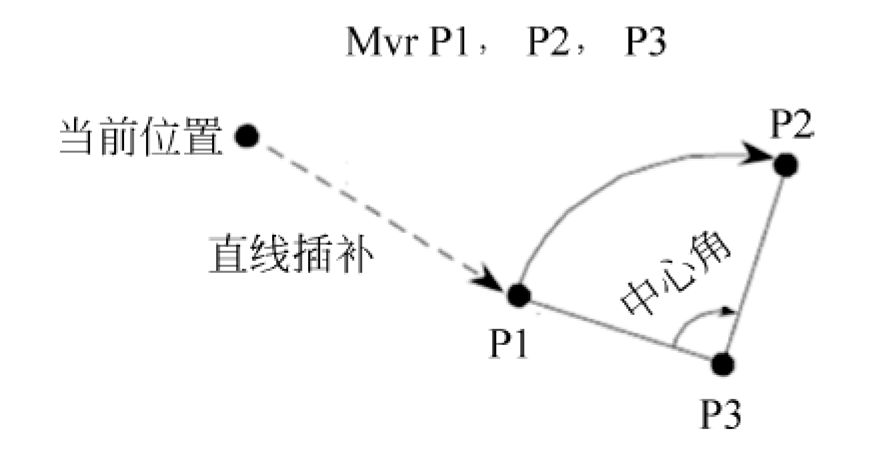


图 9-3 Mvr3 指令的运动轨迹

2. 指令格式

```
Mvr3 <起点>, ,<终点>,<圆心点> 轨迹类型,插补类型 附随语句
```

3. 术语说明

(1) 起点：圆弧起点。

(2) 终点：圆弧终点。

(3) 圆心点：圆心。

(4) 轨迹类型：常数 1＝1,绕行；常数 1＝0,捷径运行。

(5) 插补类型：常数＝0,关节插补；常数＝1,直交插补；常数＝2,通过特异点。

4. 指令例句

1 Mvr3 P1,P2,P3'——以 P1,P2,P3 点做三点型圆弧插补。

2 Mvr3 P1,J2,P3'——以 P1,J2,P3 点做三点型圆弧插补。

3 Mvr3 P1,P2,P3 Wth M_Out(17) = 1'——以 P1,P2,P3 点做三点型圆弧插补。同时指令输出信号 17 = ON。

4 Mvr3 P3,(Plt 1,5),P4 WthIf M_In(20) = 1,M_Out(21) = 1'——以 P3, (Plt 1,5),P4 点做三点型圆弧插补,同时如果输入信号 20 = ON,则指令输出信号 21 = ON。

9.4 M-V 编程语言

9.4.1 MELFA-BASIC V 的详细规格

MELFA-BASIC V 是三菱工业机器人使用的编程语言,各品牌机器人所使用的编程语言大同小异,学会一种编程语言,再学习使用其他的编程语言就很方便了。在学习使用 MELFA-BASIC V 之前,需要学习 MELFA-BASIC V 的相关知识。

1. 程序名

"程序名"只可以使用英文大写字母及数字,长度为 12 个字母。如果要使用"程序选择"功能时,则必须只使用数字作为"程序名"。

2. 指令

指令由以下部分构成:

1 Mov P1 Wth M_Out(17)=1
① ② ③ ④

① 步序号或称为"程序行号"。

② 指令。

③ 指令执行的对象:变量或数据。

④ 附随语句。

3. 变量

编程语言中对指令执行的对象使用了大量的变量。

1) 变量大分类

机器人系统中使用的变量可以进行大分类,如图 9-4 所示。

系统变量:系统变量值有系统反馈的,表示系统工作状态的变量。变量名称和数据类

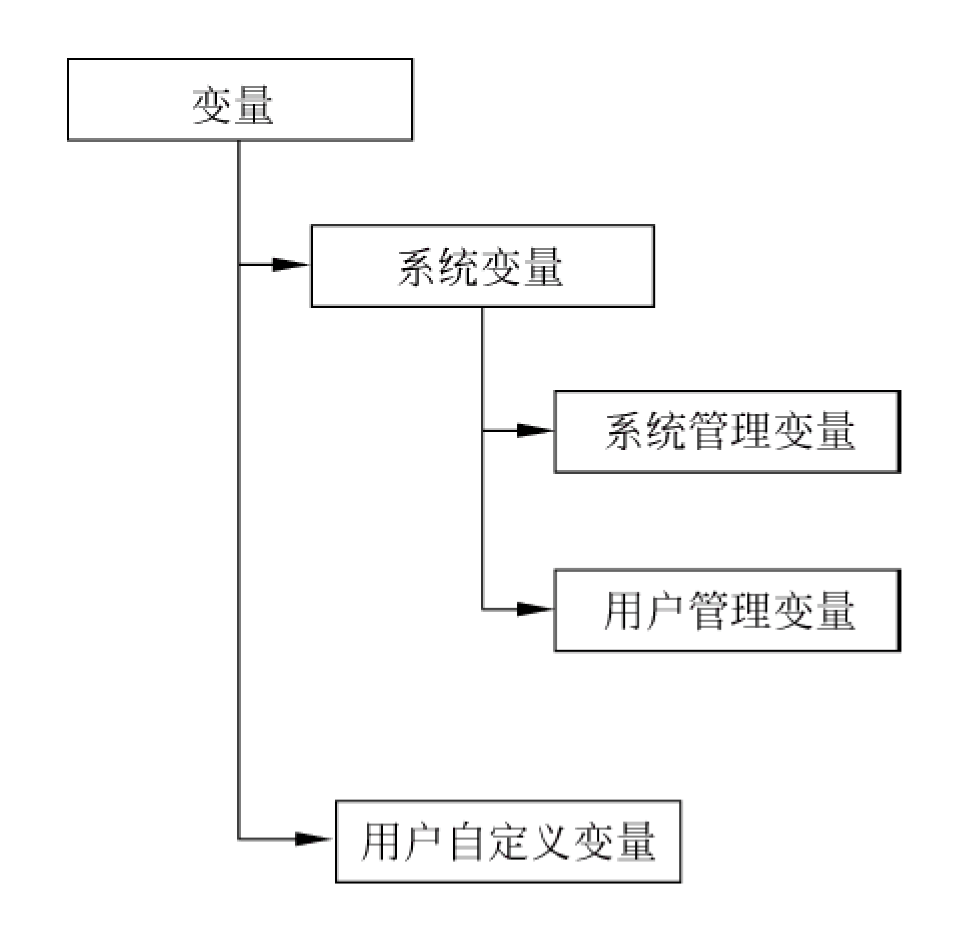


图 9-4 变量大分类

型都是预先规定了的。

系统管理变量：表示系统工作状态的变量，在自动程序中只用于表示“系统工作状态”，例如当前位置 P_CURR。

用户管理变量：是系统变量的一种，但是用户可以对其处理，例如输出信号 M_OUT(18)=1。用户在自动程序中可以指令输出信号 ON/OFF。

用户自定义变量：这类变量的名称及使用场合由用户自行定义，是使用最多的变量类型。

2) 用户变量的分类

位置变量：表示直交型位置数据，以 P 开头，例如 P1、P20。

关节型变量：表示关节型位置数据(各轴的旋转角度)，以 J 开头，例如 J1、J10。

数值型变量：表示数值，以 M 开头，例如 M1、M5(M1=0.345，M5=256)。

字符串变量：表示字符串，在变量名后加 $，例如 C1 $ =“OPENDOOR”，即变量 C1 $ 表示的是字符串“OPENDOOR”。

4. 程序【正文】

构成程序的最小单位，即指令及数据。

Mov　P1 附随语句

Mov——指令。

P1——数据。

```
1  Mov P1 Wth M_Out(17) = 1
```

Wth M_Out(17)=1 为附随语句，表示在移动指令的同时，执行输出 M_Out(17)=1。

【程序行号】

编程时，软件自动生成“程序行号”。但是 GOTO 指令、GOSUb 指令不能直接指定行号，否则报警。

【标签(指针)】

标签是程序分支的标记，用 * +英文字母构成。例如：

```
GoTo *LBL
```

…

其中，* LBL 就是程序分支的标记。

9.4.2 有特别定义的文字

1. 英文大小写

程序名、指令均可大小写，无区别。

2. 下画线(_)

下画线标注全局变量。全局变量是全部程序都可以使用的变量。在变量的第二个字母位置用下画线表示时，这种类型变量即为全局变量，例如 P_Curr，M_01，M_ABC。

3. 撇号(')

撇号表示后面的文字为注释。例如：

```
100 Mov P1 'TORU
```

其中，TORU 就是注释。

4. 星号(*)

在程序分支处做标签时，必须在第一位加星号(*)，例如 200　*KAKUNIN。

5. 逗号(,)

逗号用于分隔参数、变量中的数据，例如 P1=(200，150，…)。

6. 句号(。)

句号用于标识小数、位置变量、关节变量中的成分数据。

例如 M1=P2.X 用于标识 P2 中的 X 数据。

7. 空格

(1) 在字符串及注释文字中，空格是有文字意义的；

(2) 在行号后，必须有空格；

(3) 在指令后，必须有空格；

(4) 数据划分，必须有空格。

在指令格式中，“　”表示必须有空格。

9.4.3 数据类型

1. 字符串常数

用双引号括起来的文字部分即“字符串常数”。例如"ABCDEFGHIJKLMN"、"123"。

2. 位置数据结构

位置数据包括坐标轴，形位轴，附加轴及结构标志数据，如图 9-5 所示。

(1) X，Y，Z：表示机器人“控制点”在直角坐标系中的坐标。

(2) A/B/C：表示以机器人“控制点”为基准的机器人本体绕 XYZ 轴旋转的角度，称为“形位”。

(3) L1/L2：表示附加轴运行数据。

(4) FL1：构造标志，表示控制点与特定轴线之间的相对关系。

(5) FL2：构造标志，表示各轴的旋转角度。

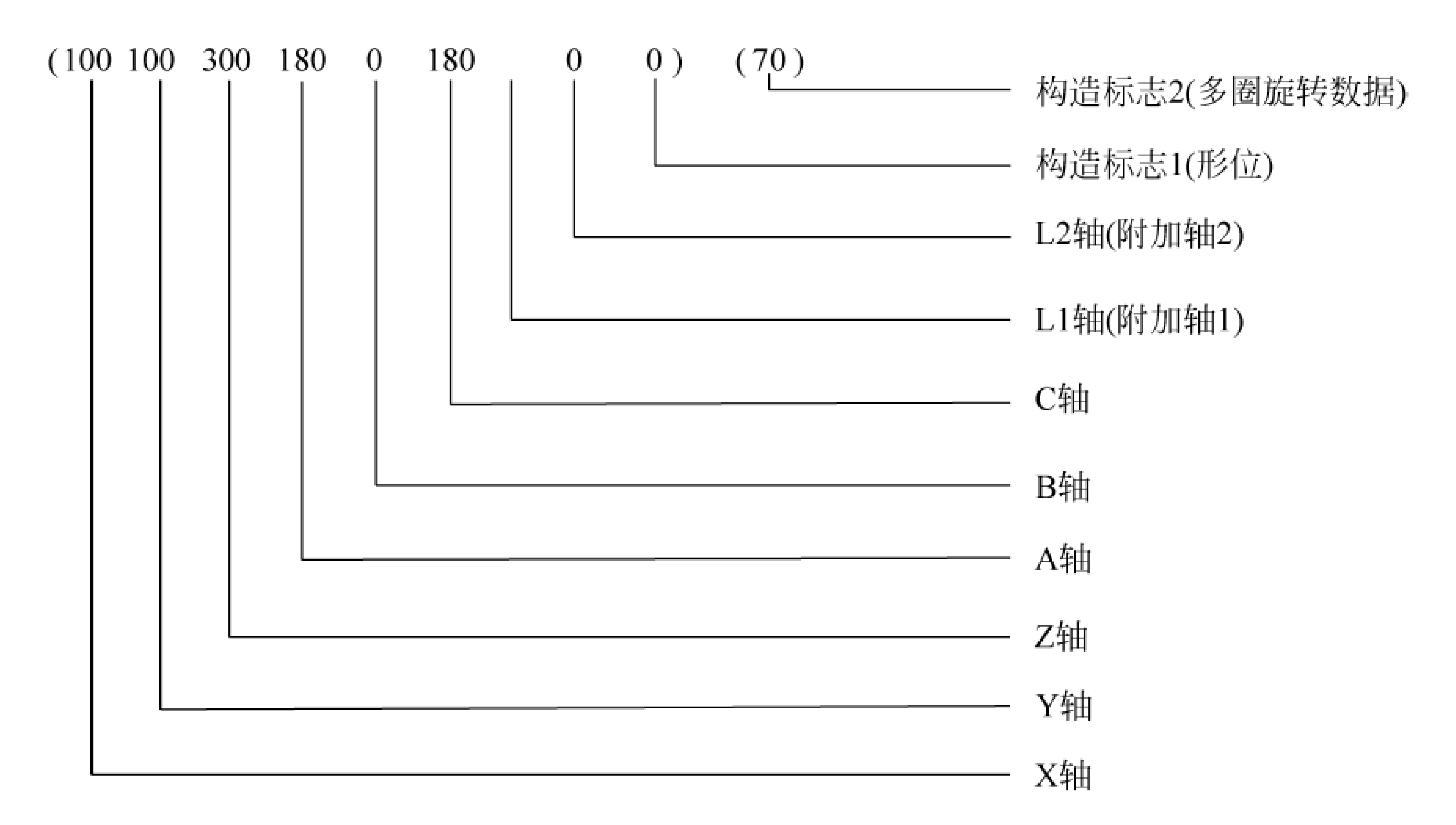


图 9-5 位置数据结构

9.5 需要思考的问题

(1) 两点型圆弧插补指令的轨迹通过参考点吗?
(2) Mvr3 指令的三个点都在圆弧轨迹上吗?
(3) 什么是附随语句?
(4) 什么是位置变量和数据变量?
(5) 什么是全局变量?
(6) 位置数据中的结构标志 FL2 是如何定义的?

第 10 章 第 10 日——编程指令的学习和使用(2)

【学习目的】

码垛是机器人经常应用的一种功能,机器人编程指令中有专用的码垛指令。本章将学习码垛指令和连续轨迹运行指令,再结合第 21 章的学习,可以对码垛功能有深入的了解。

10.1 码 垛 指 令

1. 功能

PALLET 指令也翻译为“托盘”指令、“码垛”指令,实际上是一个计算矩阵方格中各“点位中心”(位置)的指令,该指令需要设置矩阵方格有几行几列、起点终点、对角点位置、计数方向。由于该指令通常用于码垛动作,所以也就被称为“码垛”指令。

2. 指令格式

Def Plt——定义“托盘结构”指令(定义一个矩阵结构)。

```
Def  Plt  <托盘号>  <起点>  <终点 A>  <终点 B>  [<对角点>]  <列数 A>  <行数 B>  <托盘类型>
```

Plt——指定托盘中的某一点。

3. 指令样例 1

托盘的定义及类型 1 如图 10-1 所示。

```
1  Def Plt 1,P1,P2,P3,  ,3,4,1'—— 3 点型托盘定义指令
2  Def Plt 1,P1,P2,P3,P4,3,4,1'—— 4 点型托盘定义指令
```

(3 点型托盘定义指令—指令中只给出起点、终点 A、终点 B;

4 点型托盘定义指令—指令中给出起点、终点 A、终点 B、对角点)。

4. 说明

(1) 托盘号:系统可设置 8 个托盘。本数据设置第几号托盘。

(2) 起点/终点/对角点:如图 10-1 所示,用“位置点”设置。

(3) <列数 A>:起点与终点 A 之间的列数。

(4) <行数 B>:起点与终点 B 之间的行数。

(5) <托盘类型>:设置托盘中“各位置点”分布类型。

```
1 = Z 字型   2 = 顺排型   3 = 圆弧型
11 = Z 字型   12 = 顺排型   13 = 圆弧型
```

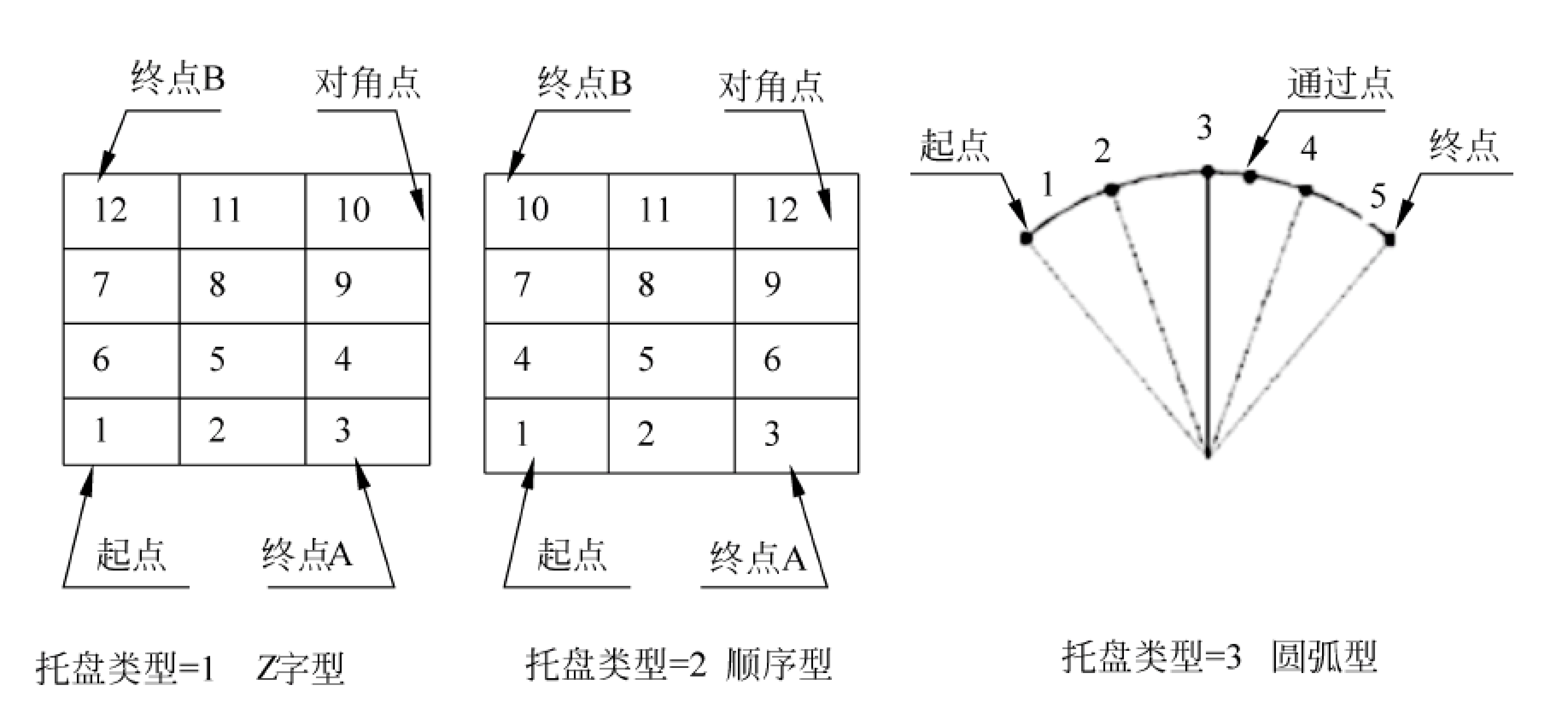


图 10-1 托盘的定义及类型 1

5. 指令样例 2

托盘的定义及类型 2 如图 10-2 所示。

(1) Def Plt 1,P1,P2,P3,P4,4,3,1'——定义 1 号托盘。4 点定义。4 列×3 行。Z 字型格式。

(2) Def Plt 2,P1,P2,P3,,8,5,2'——定义 2 号托盘。3 点定义。8 列×5 行。顺排型格式(注意 3 点型指令在书写时在终点 B 后有两个逗号)。

(3) Def Plt 3,P1,P2,P3,,5, 1,3'——定义 3 号托盘。3 点定义。圆弧型格式(注意 3 点型指令在书写时在终点 B 后有两个逗号)。

(4) (Plt 1,5)'——1 号托盘第 5 点。

(5) (Plt 1,M1)'——1 号托盘第 M1 点(M1 为变量)。

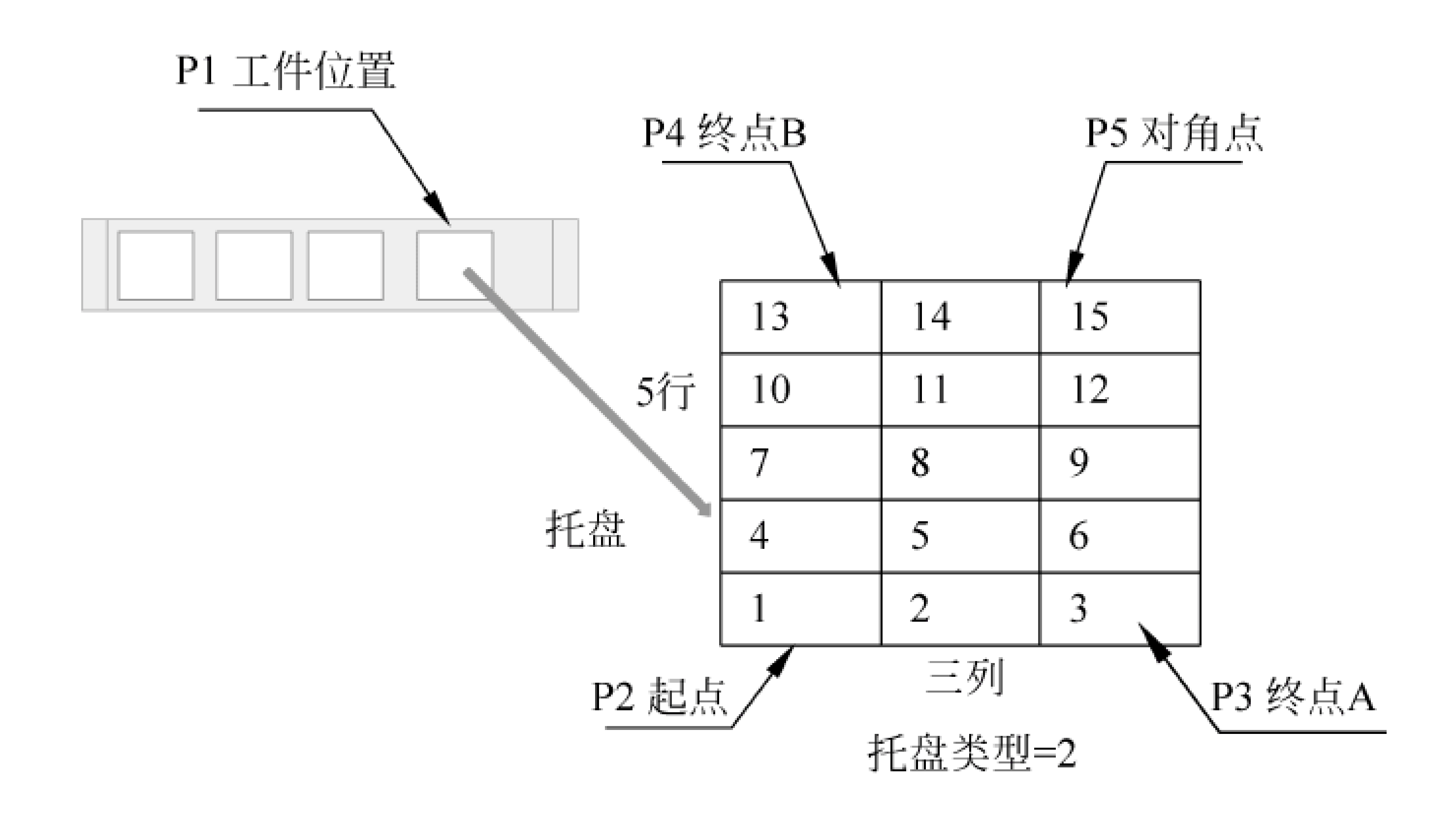


图 10-2 托盘的定义及类型 2

6. 程序样例 1

```
1  P3.A = P2.A'——设定"形位"P3 点 A 轴角度 = P2 点 A 轴角度。
2  P3.B = P2.B '——设定 P3 点 B 轴角度 = P2 点 B 轴角度。
3  P3.C = P2.C '——设定 P3 点 C 轴角度 = P2 点 C 轴角度。
4  P4.A = P2.A '——设定 P4 点 A 轴角度 = P2 点 A 轴角度。
```

5 P4.B = P2.B '——设定 P4 点 B 轴角度 = P2 点 B 轴角度。
6 P4.C = P2.C '——设定 P4 点 C 轴角度 = P2 点 C 轴角度。
7 P5.A = P2.A '——设定 P5 点 A 轴角度 = P2 点 A 轴角度。
8 P5.B = P2.B '——设定 P5 点 B 轴角度 = P2 点 B 轴角度。
9 P5.C = P2.C '——设定 P5 点 C 轴角度 = P2 点 C 轴角度。
10 Def Plt 1,P2,P3,P4,P5,3,5,2'——设定 1 号托盘,3×5 格,顺排型。
11 M1 = 1'——设置 M1 变量。
12 *LOOP'——循环指令 LOOP。
13 Mov P1, -50'——前进到 P1 点近点。
14 Ovrd 50 '——设置速度倍率。
15 Mvs P1 '——前进到 P1 点。
16 HClose 1'——1#抓手闭合。
17 Dly 0.5'——暂停。
18 Ovrd 100 '——设置速度倍率。
19 Mvs , -50'——退回到 P1 点近点。
20 P10 = (Plt 1,M1)——'定义 P10 点为 1 号托盘"M1"点,M1 为变量。
21 Mov P10, -50'——前进到 P10 点近点。
22 Ovrd 50 '——设置速度倍率。
23 Mvs P10'——运行到 P10 点。
24 HOpen 1'——打开抓手 1。
25 Dly 0.5'——暂停。
26 Ovrd 100 '——设置速度倍率。
27 Mvs, -50'——退回到 P10 点近点。
28 M1 = M1 + 1'—— M1 做变量运算
29 If M1 <= 15 Then *LOOP'——循环指令判断条件。如果 M1 小于等于 15,则继续循环。根据此循环完成对托盘 1 所有"位置点"的动作。
30 End'——结束。

7. 程序样例 2

形位在±180°附近的状态。

1 If Deg(P2.C)< 0 Then GoTo *MINUS'——如果 P2 点 C 轴角度小于 0 就跳转到 Level MINUS 行。
2 If Deg(P3.C)< -178 Then P3.C = P3.C + Rad(+360) '——如果 P3 点 C 轴角度小于 -178°就指令 P3 点 C 轴 +360°。
3 If Deg(P4.C)< -178 Then P4.C = P4.C + Rad(+360) '——如果 P4 点 C 轴角度小于 -178°就指令 P4 点 C 轴 +360°。
4 If Deg(P5.C)< -178 Then P5.C = P5.C + Rad(+360) '——如果 P5 点 C 轴角度小于 -178°就指令 P5 点 C 轴 +360°。
5 GoTo *DEFINE'跳转到 Level DEFINE 行。
6 *MINUS'——程序分支标志。
7 If Deg(P3.C)> +178 Then P3.C = P3.C - Rad(+360) '——如果 P3 点 C 轴角度大于 178°就指令 P3 点 C 轴减 360°。
8 If Deg(P4.C)> +178 Then P4.C = P4.C - Rad(+360) '——如果 P4 点 C 轴角度大于 178°就指令 P4 点 C 轴减 360°。
9 If Deg(P5.C)> +178 Then P5.C = P5.C - Rad(+360) '——如果 P5 点 C 轴角度大于 178°就指令 P5 点 C 轴减 360°。
10 *DEFINE'——程序分支标志。
11 Def Plt 1,P2,P3,P4,P5,3,5,2'——定义 1#托盘。3×5 格。顺排型。
12 M1 = 1'—— M1 为变量。
13 *LOOP'——循环指令标志。
14 Mov P1, -50'——前进到 P1 点近点。

15 Ovrd 50'——设置速度倍率。
16 Mvs P1'——前进到 P1 点。
17 HClose 1'——1 号抓手闭合。
18 Dly 0.5'——暂停。
19 Ovrd 100'——设置速度倍率。
20 Mvs, - 50'——后退到 P1 点近点。
21 P10 = (Plt 1??M1)'——定义 P10 点(为 1 号托盘中的 M1 点。M1 为变量)。
22 Mov P10, - 50'——前进到 P10 点近点。
23 Ovrd 50'——设置速度倍率。
24 Mvs P10 '——前进到 P10 点。
25 HOpen 1 '——打开抓手 1。
26 Dly 0.5 '——暂停。
27 Ovrd 100 '——设置速度倍率。
28 Mvs , - 50'——后退到 P10 点近点。
29 M1 = M1 + 1'——变量 M1 运算。
30 If M1 < = 15 Then * LOOP'——循环判断条件,如果 M1 小于等于 15,则继续循环。执行 15 个点的抓取动作。
31 End'——结束。

10.2 连续轨迹运行指令 CNT

1. 功能

连续轨迹运行是指机器人控制点在运行通过各位置点时,不做每一点的加减速运行,而是以一条连续的轨迹通过各点,如图 10-3 所示。

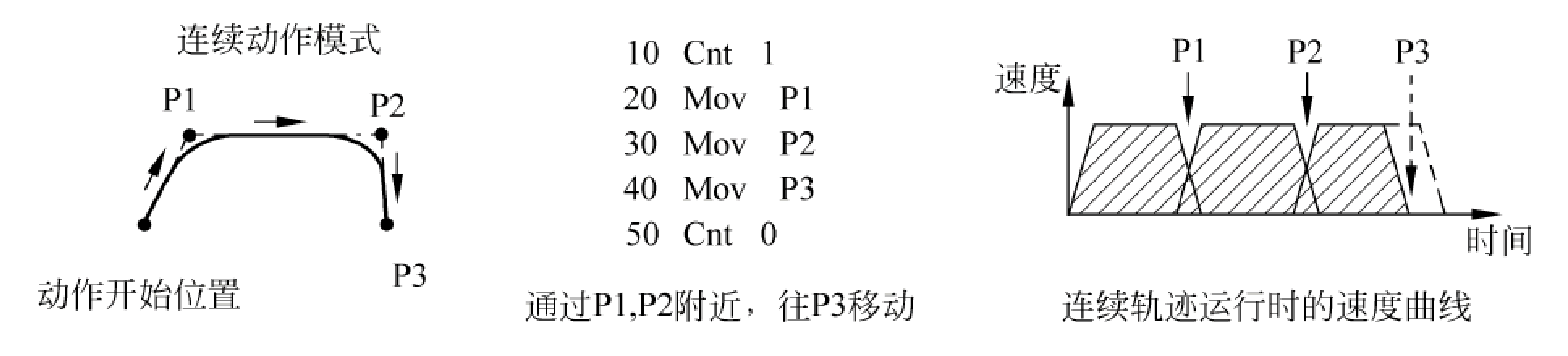


图 10-3 连续轨迹运行时的运行轨迹和速度曲线

2. 指令格式

Cnt <1/0>[,<数值 1>][,<数值 2>]

说明:

<1/0>: Cnt 1——连续轨迹运行。

Cnt 0——连续轨迹运行无效。

<数值 1>: 过渡圆弧尺寸 1。

<数值 2>: 过渡圆弧尺寸 2。

在连续轨迹运行通过某一位置点时,其轨迹不实际通过位置点,而是一个过渡圆弧,这个过渡圆弧轨迹由指定的数值构成。

3. 程序样例

1 Cnt 0'——连续轨迹运行无效。

```
2  Mvs P1'——移动到 P1 点。
3  Cnt 1'——连续轨迹运行有效。
4  Mvs P2'——移动到 P2 点
5  Cnt 1,100,200'——指定过渡圆弧数据 100mm/200mm。
6  Mvs P3'——移动到 P3 点。
7  Cnt 1,300'——指定过渡圆弧数据 300mm/300mm。
8  Mov P4'——移动到 P4 点。
9  Cnt 0'——连续轨迹运行无效。
10 Mov P5'——移动到 P5 点。
```

4. 说明

(1) 从 Cnt1 到 Cnt0 的区间为连续轨迹运行有效区间。

(2) 系统初始值为 Cnt0(连续轨迹运行无效)。

(3) 如果省略"数值 1""数值 2"的设置,其过渡圆弧轨迹如图 10-4 中虚线所示,圆弧起始点为"减速开始点",圆弧结束点为"加速结束点"。

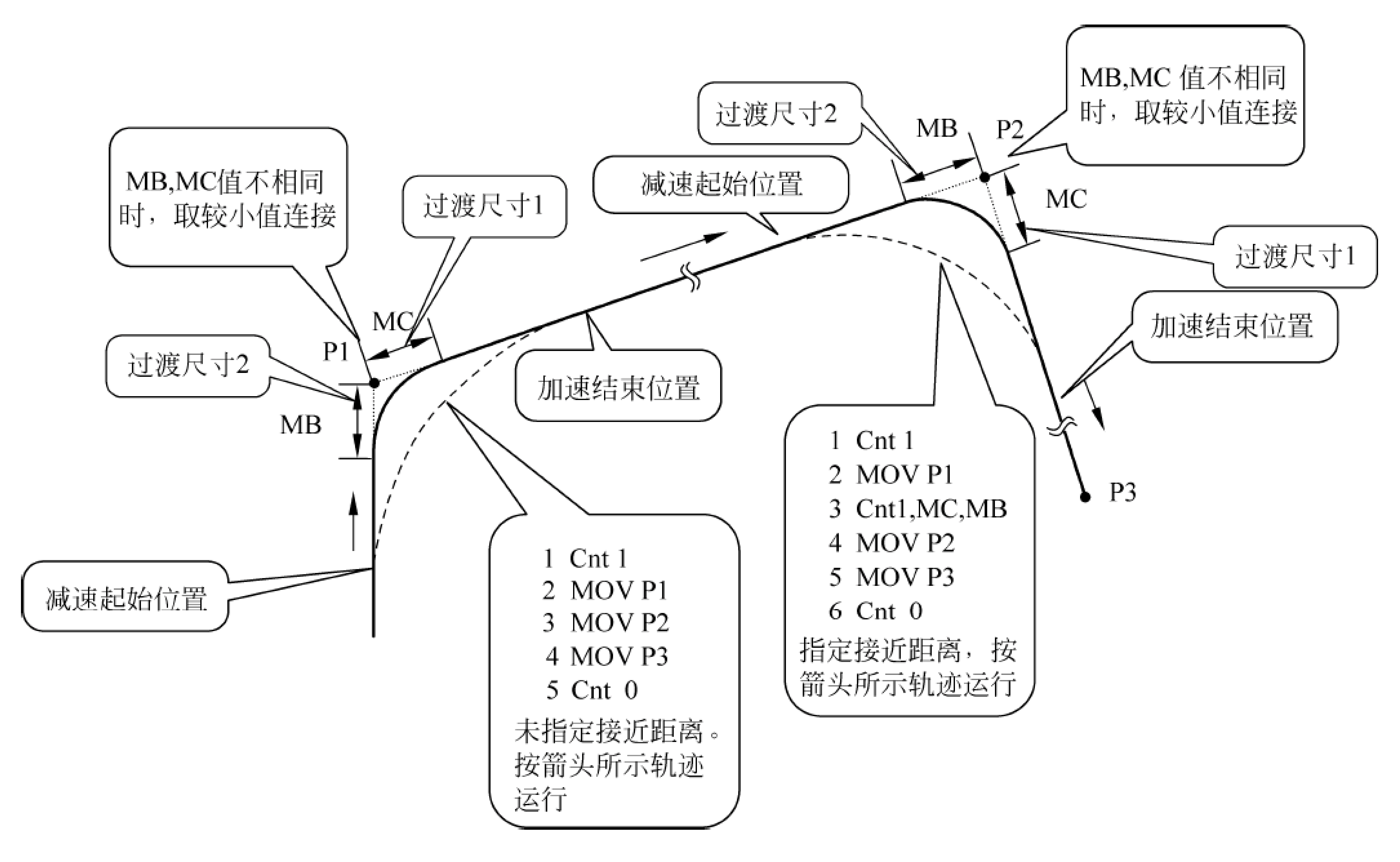


图 10-4　连续运行轨迹及过渡尺寸

10.3　需要思考的问题

(1) 码垛指令的实质是什么?

(2) 3 点型码垛指令与 4 点型码垛指令有何区别?

(3) 托盘上各点的分布有几种方式?

(4) 如何连续执行对托盘上各点的定位?

(5) 连续轨迹指令的运行轨迹是否通过各位置点?

(6) 连续轨迹指令的运行是否会出现反复加减速?

第 11 章 第 11 日——机器人的“控制点”及“位置点”数据运算

【学习目的】

在第 3 章的学习中，已经让机器人动了起来。在第 6 章的学习中，已经让机器人按简单的运动程序自动运行，但是我们只看到机器人的运行，没有明确意识到机器人的工作点是哪一点。如果要求机器人到达某一位置点，是要求机器人的某一个部位“点”到达“位置点”，通过本章的学习要明确这些问题。

11.1 机器人的“控制点”

对初学者而言，机器械人的“控制点”在哪里，这是一个既明白又糊涂的问题。因为用示教单元操作时，可以看见机器人在运行，但对精确定位时是要求机器人的哪一个部位到达指定位置又有些糊涂。实际上，“机器人控制点”是指“机器人本体”上的一个点，在出厂时，这个点被定义为机器人“法兰中心点”，就是机械 IF 的中心点，如图 11-1 和图 11-2 所示。

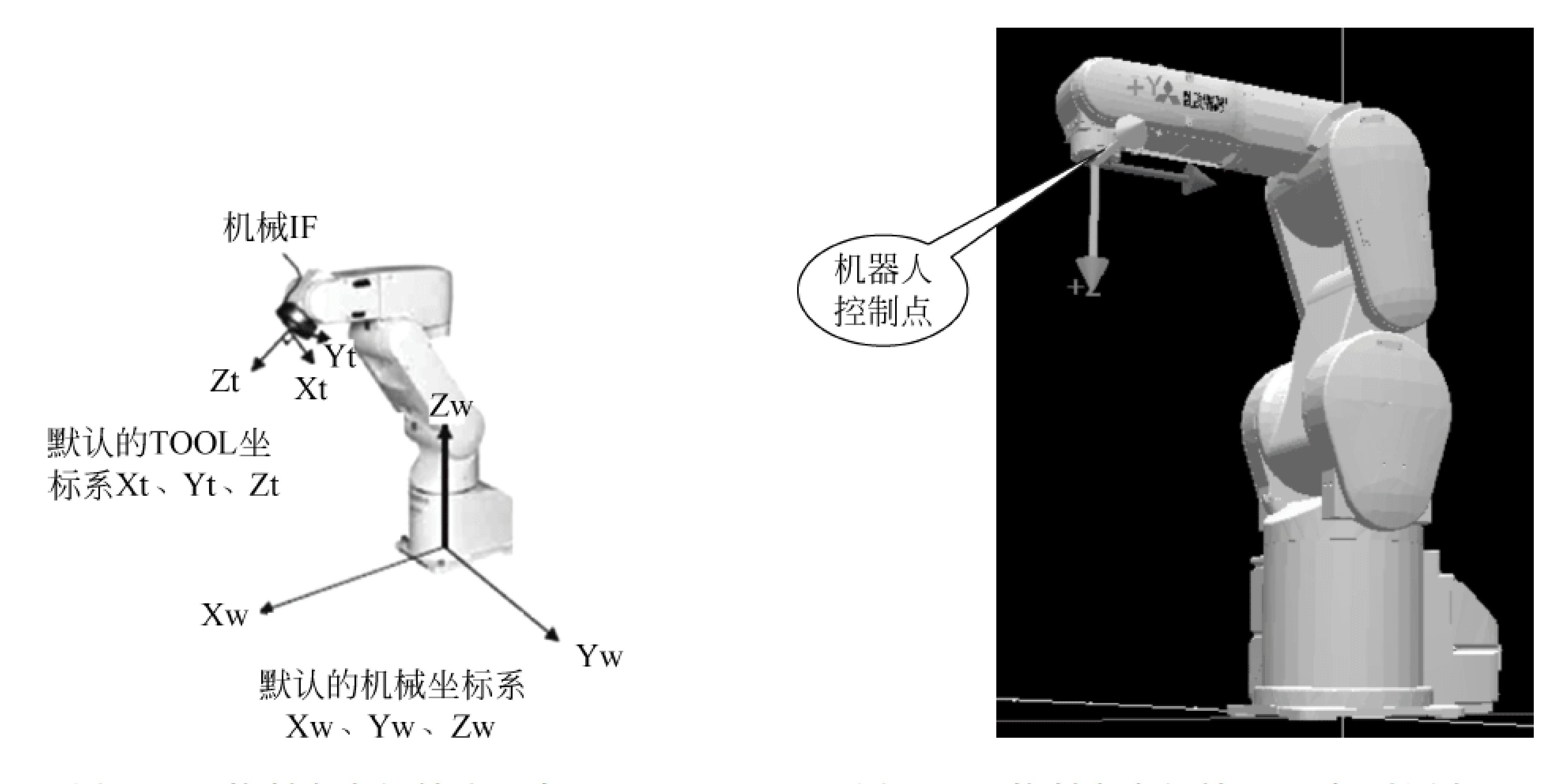


图 11-1 控制点在机械法兰中心

图 11-2 控制点在机械 IF 坐标系原点

如果设置了 TOOL 坐标系后（TOOL 坐标系即工具坐标系，大多数时候也即抓手坐标系，抓手就连接在法兰上），“机器人控制点”就是 TOOL 坐标系的原点，如图 11-3 所示，所以在 JOG 动作和自动程序中，是指令这一“机器人控制点”移动到指定的位置。

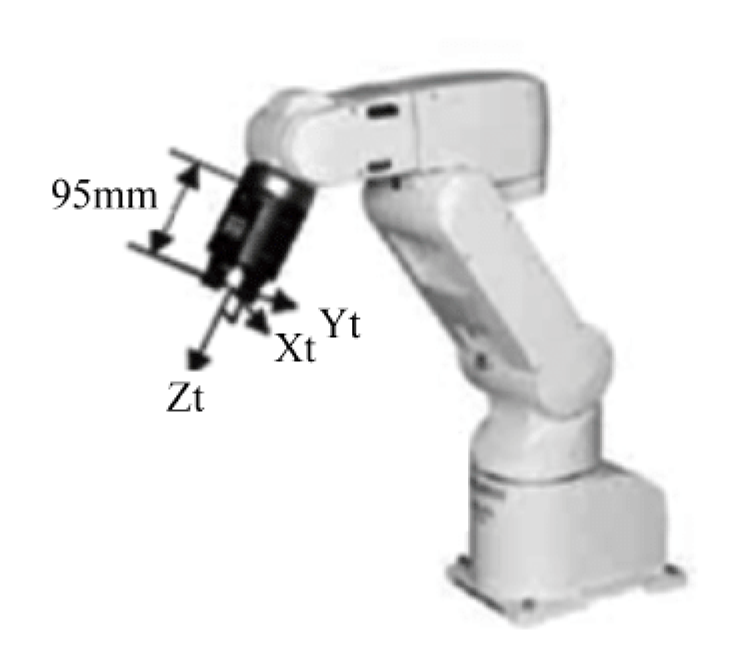


图 11-3　控制点在 TOOL 坐标系原点

11.2　如何表示一个“位置点”

“位置点”如何表示呢？确定“位置点”需要以下数据。

1. 坐标位置和旋转角度位置

如图 11-4 和图 11-5 所示，“位置点”由以下几个数据构成。

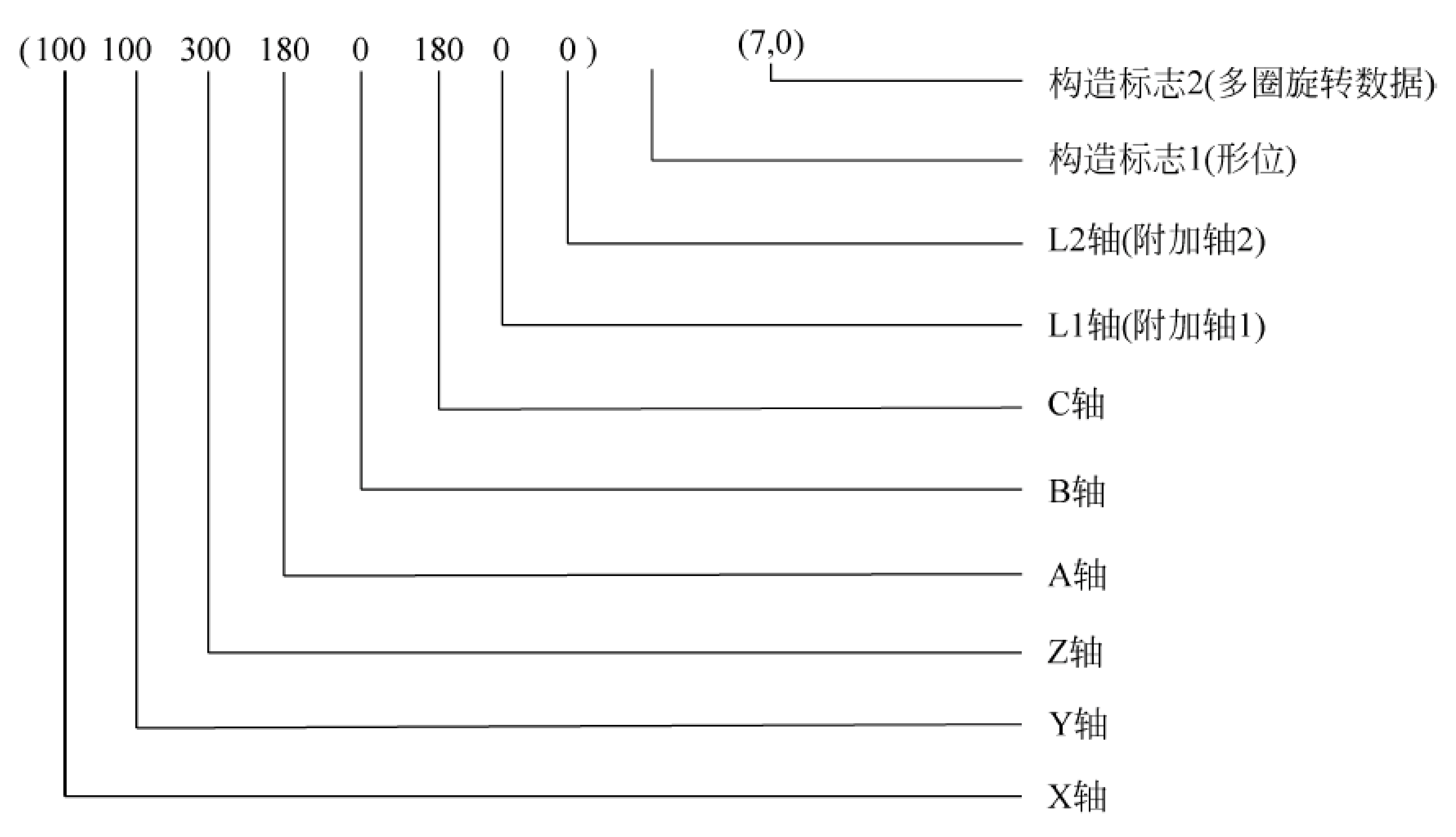


图 11-4　表示“位置点”的 8 个数据

(1) X,Y,Z——表示“机器人控制点”在直角坐标系中的坐标。

(2) A,B,C——表示绕 XYZ 轴旋转的角度(就一个“点”位而言，没有旋转的概念，所以旋转是指以该“位置点”为基准，以抓手为刚体，绕世界坐标系的 XYZ 轴旋转。这样即使是同一个“位置点”，抓手的形位也有 N 种变化)。

注意：X,Y,Z 和 A,B,C 全部以世界坐标系为基准。

(3) L1,L2——附加轴(伺服轴)定位位置。

(4) FL1——构造标志(上下左右高低位置)。

(5) FL2——各关节轴旋转度数。

2. 结构标志

(1) FL1——结构标志(上下左右高低位置)。用一组二进制数表示，“上下左右高低”用不同的 bit 表示，如图 11-5 所示。

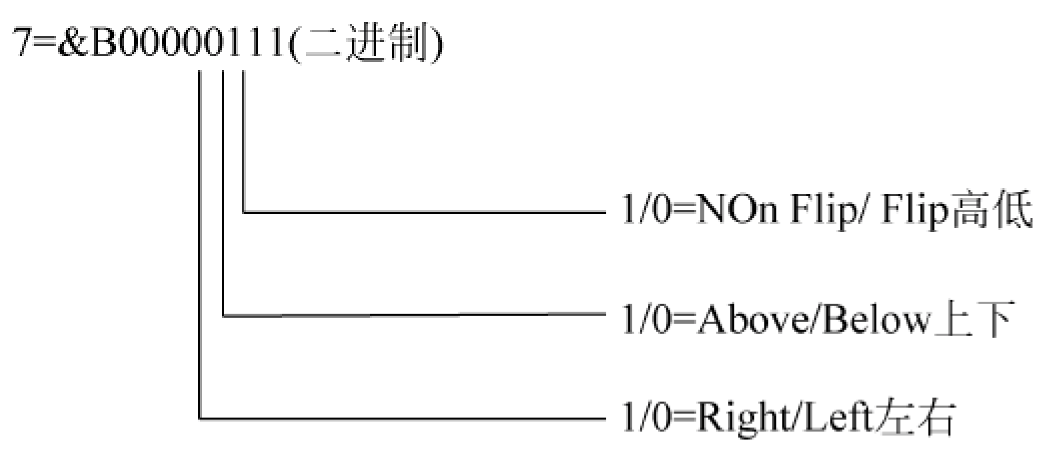


图 11-5　表示 FL1——结构标志的二进制数

（2）FL2——各关节轴旋转度数。用一组十六进制数表示，如图 11-6 所示。

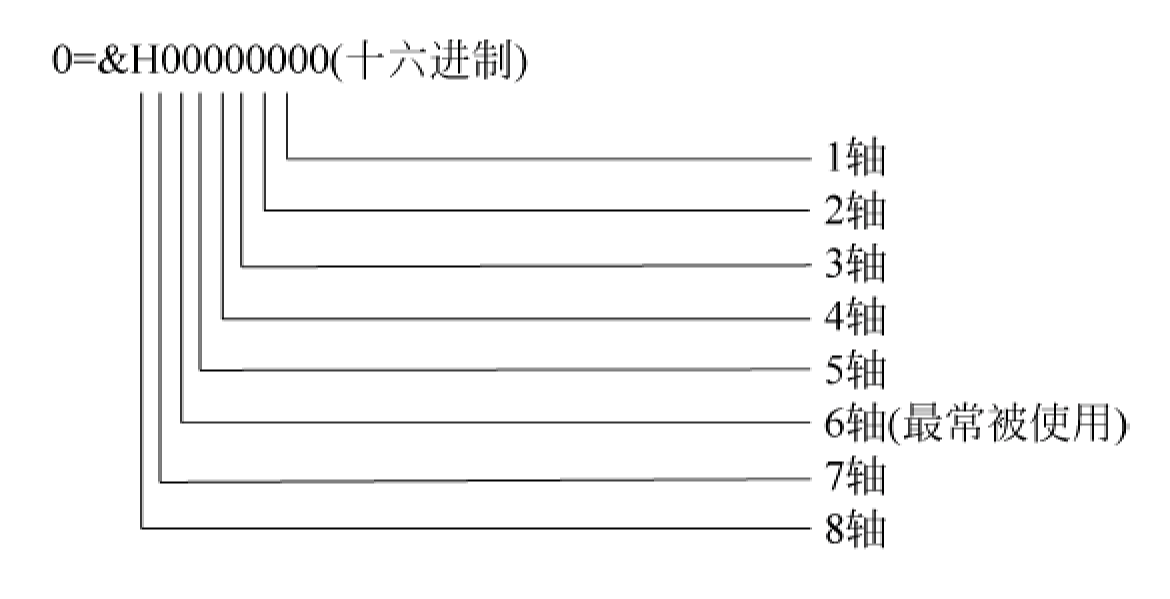


图 11-6　表示 FL2——各关节轴旋转度数的十六进制数

11.3　结构标志 FL1

“位置点”是由 X，Y，Z，A，B，C（FL1，FL2）标记的。由于机器人结构的特殊性，即使是同一“位置点”，机器人也可能出现不同的“形位”。为了区别这些“形位”，采用了结构标志。用位置标记的（X，Y，Z，A，B，C）（FL1，FL2）中的“FL1”标记。

11.3.1　垂直多关节型机器人

1. 左右标志

（1）5 轴机器人：以 J1 轴旋转中心线为基准，判别第 5 轴法兰中心点 P 位于“该中心线”的左面还是右面。如果在右边，则 FL1 bit2＝1；如果在左边，则 FL1 bit2＝0；如图 11-7 所示。

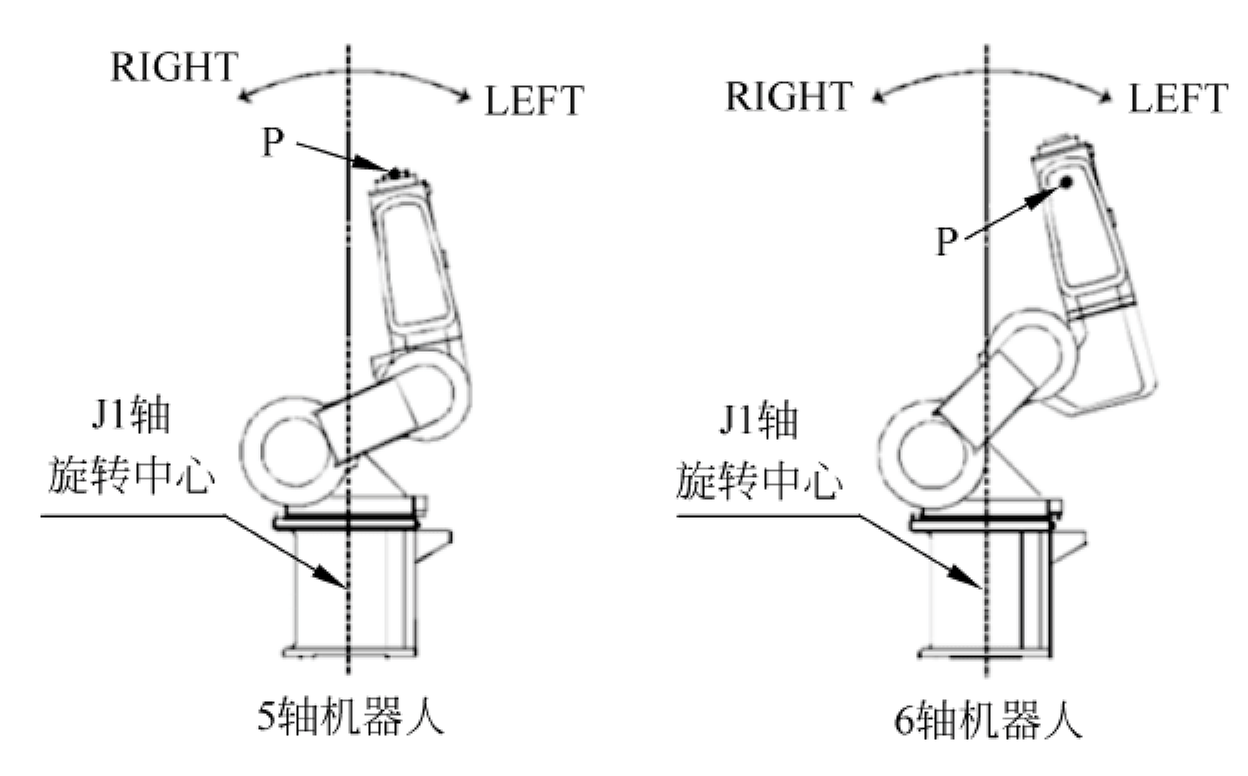


图 11-7　左右判定

(2) 6 轴机器人：以 J1 轴旋转中心线为基准，判别 J5 轴中心点 P 位于该中心线的左面还是右面。如果在右边，则 FL1 bit2＝1；如果在左边，则 FL1 bit2＝0；如图 11-8 所示。

注意：FL1 标志信号用一组二进制码表示，检验左右位置用 bit2 表示。

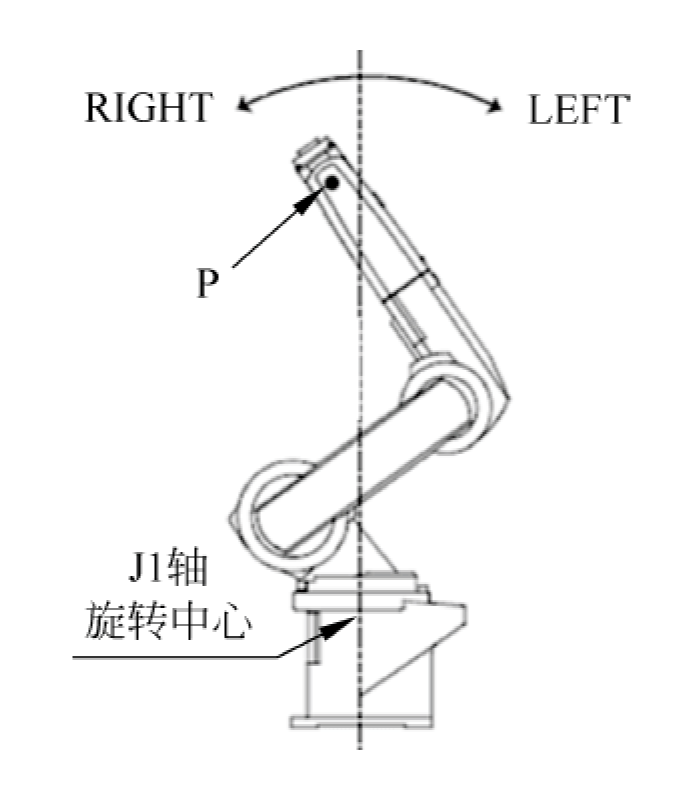


图 11-8 FL1 标志的左右判断

2. 上下判断

(1) 5 轴机器人：以 J2 轴旋转中心和 J3 轴旋转中心的连接线为基准，判别 J5 轴中心点 P 是位于该中心连接线的上面还是下面。如果在上面，则 FL1 bit1＝1；如果在下面，则 FL1 bit1＝0；如图 11-9 所示。

(2) 6 轴机器人：以 J2 轴旋转中心和 J3 轴旋转中心的连接线为基准，判别 J5 轴中心点 P 是位于该中心连接线的上面还是下面。如果在上面，则 FL1 bit1＝1；如果在下面，则 FL1 bit1＝0；如图 11-10 所示。

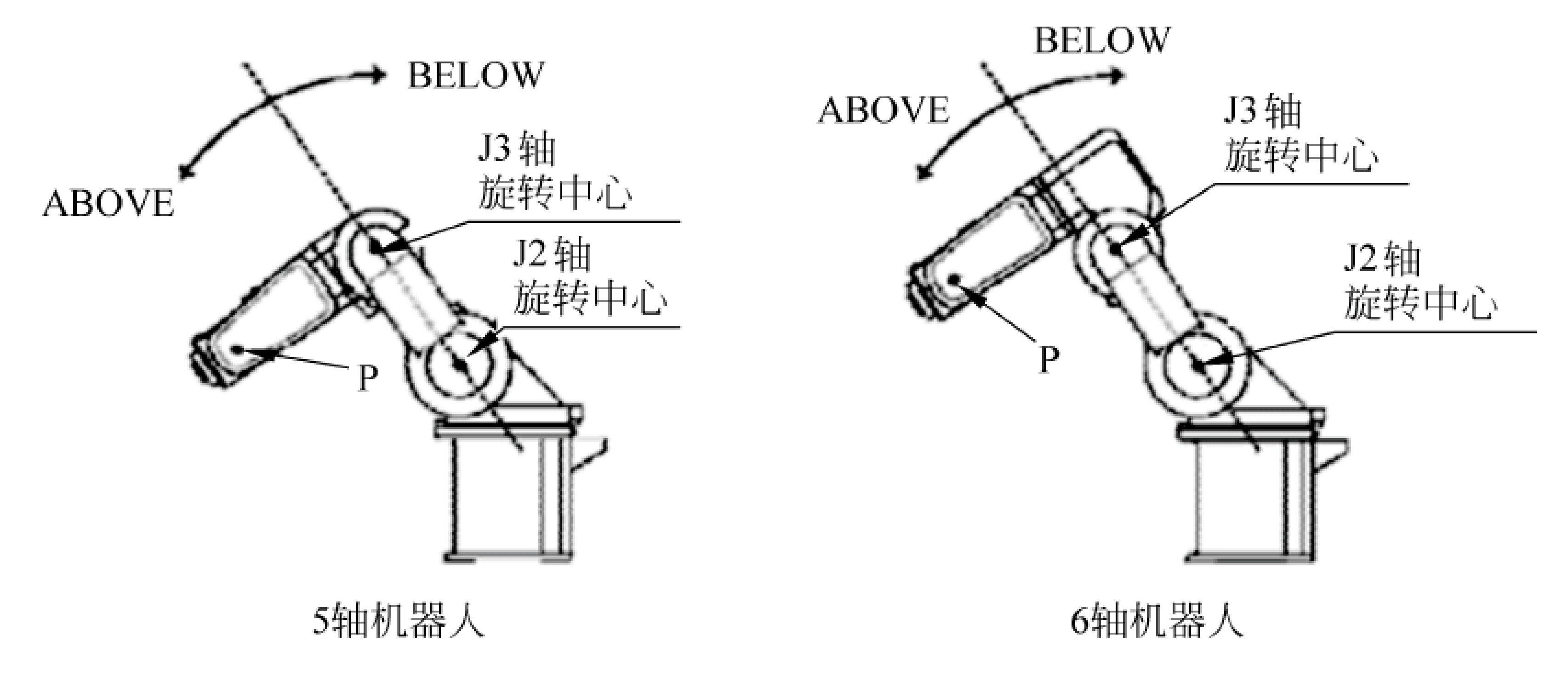


图 11-9 FL1 标志中上下的判定

注意：FL1 标志信号用一组二进制码表示，检验上下位置用 bit1 表示。

3. 高低判断

第 6 轴法兰面(6 轴机型)方位判断。以 J4 轴旋转中心和 J5 轴旋转中心的连接线为基准，判别 6 轴的法兰面是位于该中心连接线的上面还是下面。如果在下面，则 FL1 bit0＝1；如果在上面，则 FL1 bit0＝0；如图 11-11 所示。

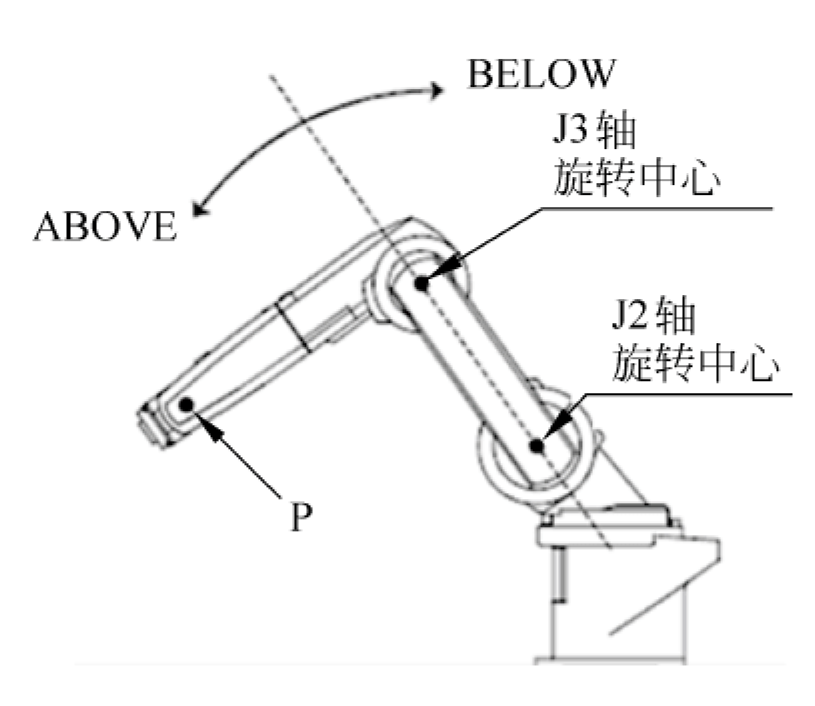


图 11-10 FL1 标志中上下的判定及显示

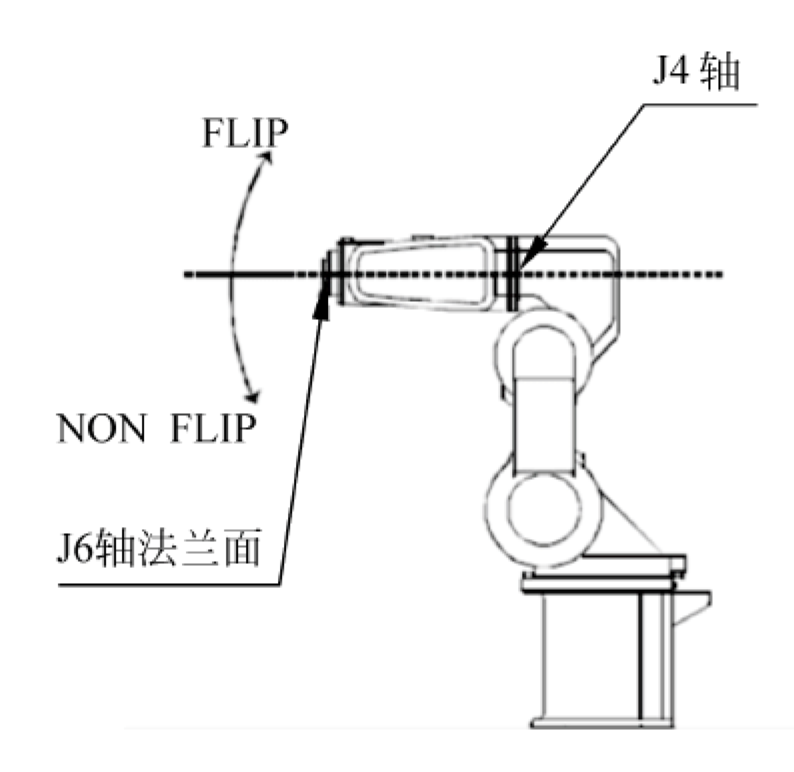


图 11-11 J6 轴法兰面位置的判定

注意：FL1 标志信号用一组二进制码表示，检验高低位置用 bit0 表示。

11.3.2 水平运动型机器人

以 J1 轴旋转中心和 J2 轴旋转中心的连接线为基准，判别机器人前端位置控制点是位于该中心连接线的左面还是右面。如果在右面，则 FL1 bit2＝1；如果在左面，则 FL1 bit2＝0；如图 11-12 所示。

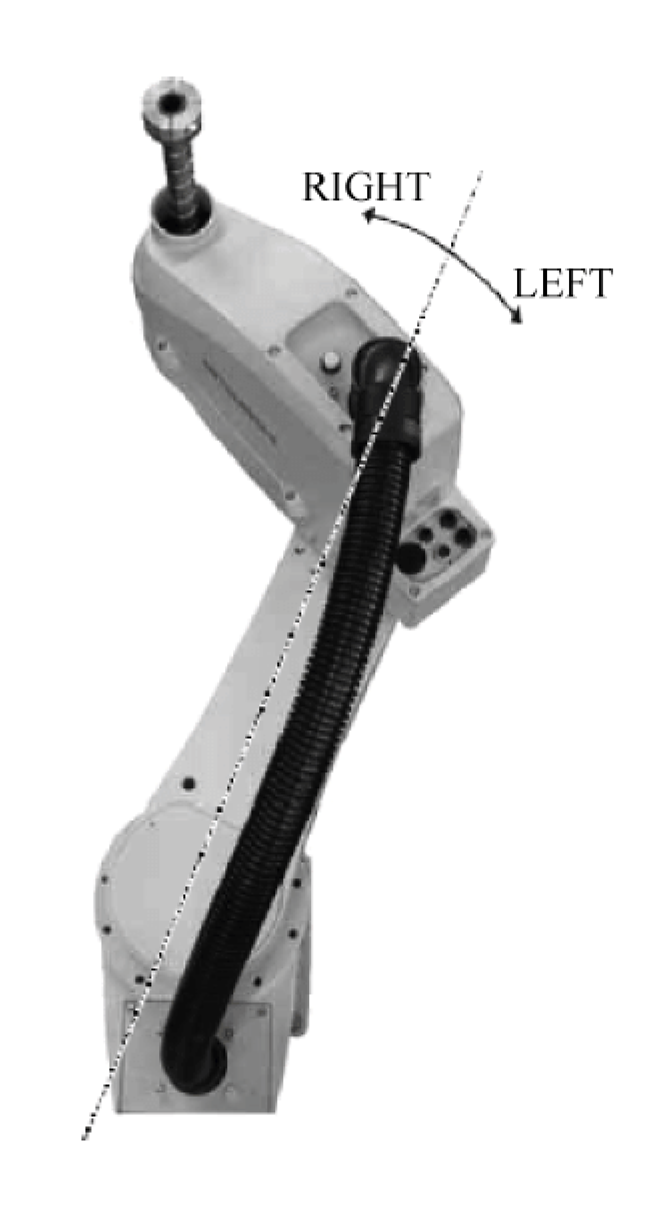


图 11-12　水平运动型机器人的 FL1 标志

11.4 结构标志 FL2

FL2 标志为各关节轴旋转度数，用一组十六进制数表示，如图 11-13 所示。

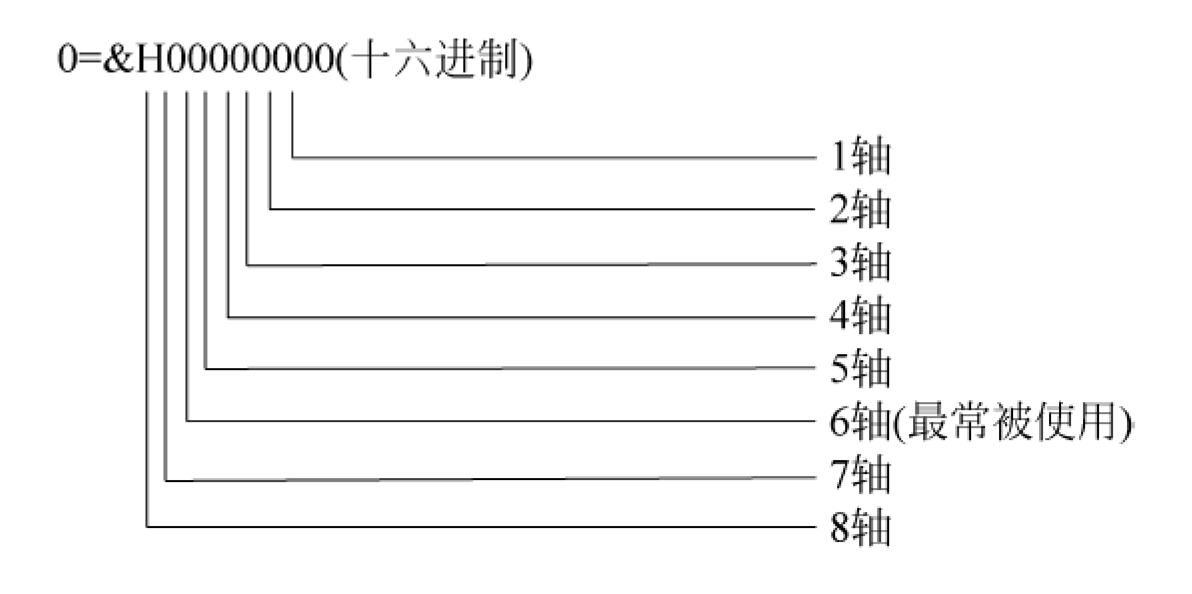


图 11-13　表示 FL2 各关节轴旋转度数的十六进制数

各轴的旋转角度与数值之间的关系如图 11-14 所示。

以 J6 轴为例：

旋转角度＝－180°～0～180°　FL2＝H00000000

旋转角度＝180°～540°　　　FL2＝H00100000

旋转角度＝540°～900°　　　FL2＝H00200000
旋转角度＝-180°～-540°　　FL2＝H00F00000
旋转角度＝-540°～-900°　　FL2＝H00E00000

各轴角度	–900~–540	–540~–180	–180~0	0~180	180~540	540~900
FL2数据	–2(E)	–1(F)	0	0	1	2

图 11-14　旋转度数与十六进制数的关系

11.5　位置点的计算方法

11.5.1　位置点乘法运算

位置数据乘法运算表达式如下：

```
P100 = P1 * P2
```

位置数据的乘法运算实际是变换到 TOOL 坐标系的过程。在下例中，“P100＝P1 * P2”，P1 点是在世界坐标系中确定的点。又将 P1 点作为 TOOL 坐标系中的原点，P2 是 TOOL 坐标系中的坐标点，如图 11-15 所示。注意 P1、P2 点的排列顺序，顺序不同，意义也不一样。

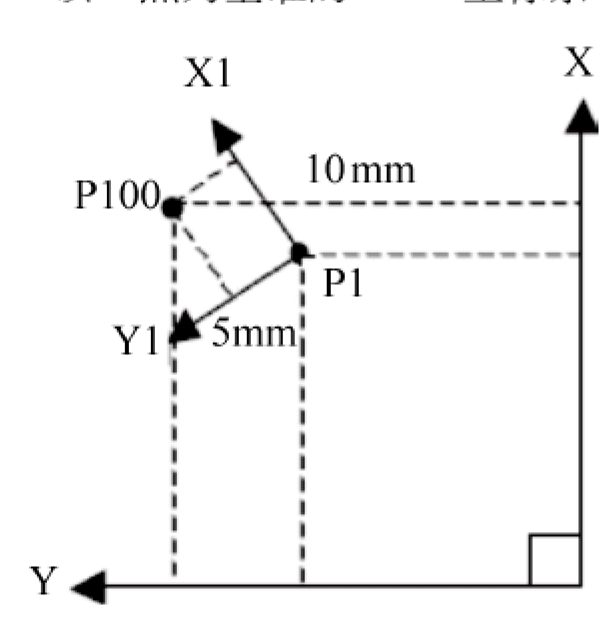


图 11-15　位置数据运算——乘法

乘法运算就是在 TOOL 坐标系中的“加法运算”，除法运算就是在 TOOL 坐标系中的“减法运算”。由于乘法运算经常使用在“根据当前点位置计算下一点的位置”，所以特别重要，使用者需要仔细体会。

程序样例：

```
P1 = (200,150,100,0,0,45)(4,0)'——P1 点数值。
P2 = (10,5,0,0,0,0)(0,0)'——P2 点数值。
P100 = P1 * P2'——P1 与 P2 的乘法运算。
Mov P1'——前进到 P1 点。
Mvs P100'——直线前进到 P100 点。
```

11.5.2　位置数据的加法/减法

位置数据的加法运算表达式如下：

```
P100 = P1 + P2
```

加法运算是以机器人世界坐标系为基准，以 P1 为起点，P2 点为坐标值进行的加法运算

(减法运算可以理解为以 P1 为起点，P2 点为坐标值进行的减法运算)，如图 11-16 所示。

样例：

```
P1 = (200,150,100,0,0,45)(4,0)'——P1 点数值。
P2 = (5,10,0,0,0,0)(0,0)'——P2 点数值。
P100 = P1 + P2'——P1 与 P2 的加法运算。
Mov P1'——前进到 P1 点。
Mvs P100'——直线前进到 P100 点。
```

因此从本质上来说，位置数据的乘法与加法的区别在于各自依据的坐标系不同，但都是以第 1 点为基准，第 2 点作为绝对值增量进行运算。

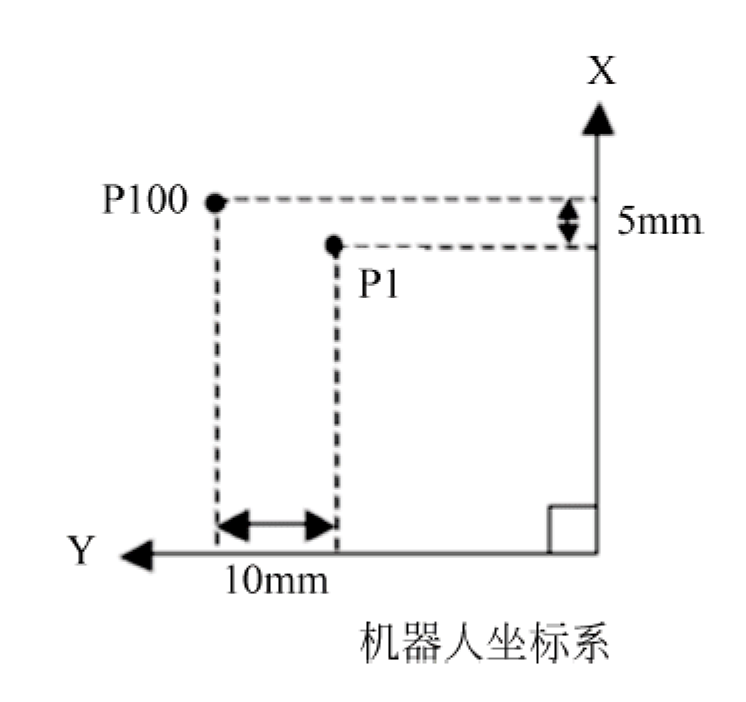


图 11-16　位置数据运算——加法

11.6　需要思考的问题

(1) 机器人控制点在机器人的哪个部位？

(2) 确定一个“位置点”需要几个数据？

(3) 只用直角坐标是否能够完整表示机器人的“形位”？

(4) 结构标志 FL1 表示什么内容？

(5) 结构标志 FL2 表示什么内容？

(6) 如何进行位置点的乘法运算？

(7) 如何进行位置点的除法运算？

第 12 章 第 12 日——编程指令的学习和使用(3)

【学习目的】

本章进行编程指令的深度学习,学习程序结构方面的指令。实际上,任何一个实用的程序都不会是简单的顺序结构,而是一个复杂的程序结构,因此在面对一个工程项目时,应该是先构建程序结构,再进行各分支程序的编程,这是学习者应该牢记的。

12.1 无条件跳转指令

GoTo 无条件跳转:只要程序运行到 GoTo,就无条件地执行跳转。GoTo 必须指定跳转的目标位置。

12.2 判断-选择指令

1. 功能

本指令是根据“条件”执行“程序分支跳转”的指令,是改变程序流程的基本指令。

2. 指令格式 1

```
If <判断条件式> Then <流程 1 >  [Else <流程 2 >]
```

这种指令格式是在程序一行里书写的判断-执行语句。如果条件成立就执行 Then 后面的程序指令;如果条件不成立则执行 Else 后面的程序指令。

3. 指令例句 1

```
1  If M1 > 10 Then  * L100'——如果 M1 大于 10,则跳转到 * L100 行。
2  If M1 > 10 Then GoTo  * L20 Else GoTo  * L30'——如果 M1 大于 10,则跳转到 * L20 行,否则跳转到
 * L30 行。
```

4. 指令格式 2

如果判断—跳转指令的处理内容较多,无法在一行程序里表示,就使用指令格式 2。

```
If <判断条件式>
Then
<流程 1 >
Else
<流程 2 >
EndIf
```

如果条件成立则执行 Then 后面一直到 Else 的程序行；如果条件不成立则执行 Else 后面到 EndIf 的程序行。EndIf 用于表示流程 2 的程序结束。

5. 指令例句 2

```
1  If M1 > 10 Then '——如果 M1 大于 10,则
2  M1 = 10'——赋值
3  Mov P1'——前进到 P1 点
4  Else'——否则
5  M1 =- 10'——赋值
6  Mov P2'——前进到 P2 点
7  EndIf'——本指令结束
```

6. 指令例句 2

多级 If…Then…Else…EndIf 嵌套：

```
1  If M1 > 10 Then'——(第 1 级判断 - 执行语句)
2  If M2 > 20 Then'——(第 2 级判断 - 执行语句)
3  M1  =  10'——赋值
4  M2  =  10'——赋值
5  Else'——否则
6  M1  =  0'——赋值
7  M2  =  0'——赋值
8  EndIf'——(第 2 级判断 - 执行语句结束)
9  Else'——否则
10 M1  =- 10'——赋值
11 M2  =- 10'——赋值
12 EndIf'——(第 1 级判断 - 执行语句结束)
```

7. 指令例句 3

在对 Then 及 Else 的流程处理中，以 Break 指令跳转到 EndIf 的下一行。从 If…Then…EndIf 的流程中跳转出来(不要使用 GoTo 指令跳转)。

```
1  If M1 > 10 Then'——(第 1 级判断 - 执行语句)
2  If M2 > 20 Then Break'——如果 M2 > 20 就跳转出本级判断执行语句(本例中为 10 行)
3  M1  =  10'——赋值
4  M2  =  10'——赋值
5  Else'——否则
6  M1  =- 10'——赋值
7  If M2 > 20 Then Break'——如果 M2 > 20 就跳转出本级判断执行语句(本例中为 10 行)
8  M2  =- 10'——赋值
9  EndIf'——结束判断指令
10 If M_BrkCq = 1 Then Hlt
11 Mov P1'——前进到 P1 点
```

8. 说明

(1) 多行型指令 If…Then…Else…EndIf 必须书写 EndIf，不得省略，否则无法确定"流程 2"的结束位置。

(2) 不要使用 GoTo 指令跳转到本指令之外。

(3) 嵌套多级指令最多为 8 级。

(4) 在对 Then 及 Else 的流程处理中，以 Break 指令跳转到 EndIf 的下一行。

12.3 选择指令 Select Case

1. 功能

本指令用于根据不同的条件选择执行不同的程序块。指令流程参看图 12-1。

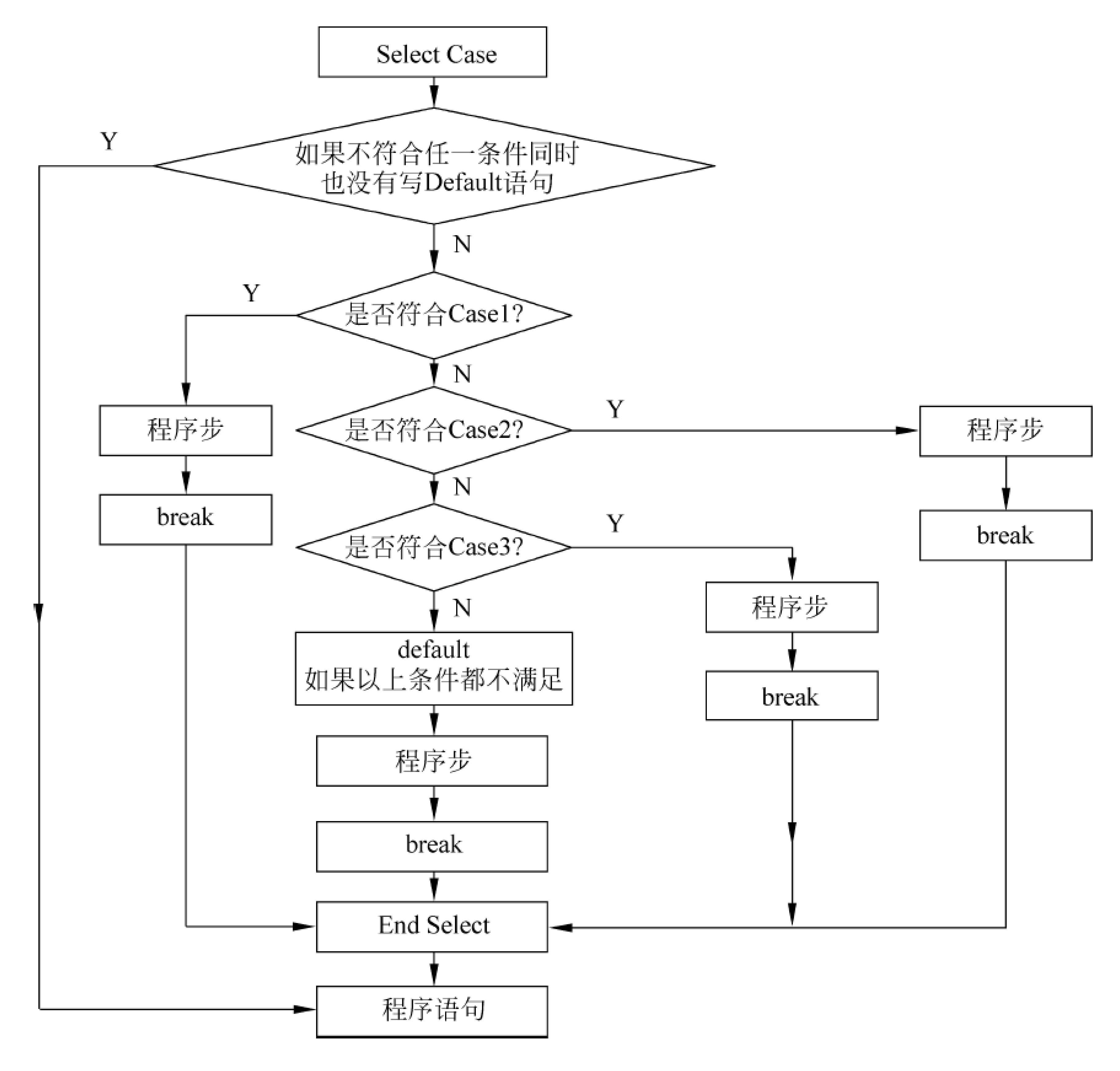


图 12-1　Select Case 语句的执行流程

2. 指令格式

```
Select  <条件>
Case  <计算式>
[<处理>]
Break
Case  <计算式>
[<处理>]
Break
Default
[<处理>]
Break
End Select
```

<条件>——数值表达式。

3. 指令例句

```
Select MCNT
M1 = 10
Case Is <=  10'——如果 MCNT≤10
Mov P1'——前进到 P1 点
Break'——跳转到程序结束
Case 11 '——如果 MCNT = 11 OR MCNT = 12
Case 12
Mov P2'——前进到 P2 点
Break'——跳转到程序结束
Case 13 To 18'——如果 13≤MCN≤18
Mov P4'——前进到 P4 点
Break'——跳转到程序结束
Default'——除上述条件以外
M_Out(10) = 1'——赋值
Break'——跳转到程序结束
End Select'——选择语句结束
```

4. 说明

(1) 如果"条件"的数据与某个 case 的数据一致,则执行到 Break 行然后跳转到 End Select 行。

(2) 如果条件都不符合,就执行 Default 规定的程序。

(3) 如果没有 Default 指令规定的程序,就跳到 End Select 的下一行。

12.4 选择指令 On GoTo

1. 功能

本指令是根据不同条件跳转到不同程序分支处的指令。判断条件是计算式,可能有不同的计算结果,根据不同的计算结果跳转到不同程序分支处。本指令与 On GoSub 指令的区别是:On GoSub 是跳转到子程序,On GoTo 指令是跳转到某一程序行。

指令流程参看图 12-2。

2. 指令格式

```
On  <条件计算式>  GoTo <程序行标签 1 ><程序行标签 2 >
```

3. 指令例句

```
On M1 GoTo  * ABC1, * LJMP, * LM1_345,  * LM1_345,  * LM1_345, * L67, * L67
  '——如果 M1 = 1,就跳转到 * ABC1 行。
  如果 M1 = 2,就跳转到 * LIMP 行。
  如果 M1 = 3,M1 = 4,M1 = 5,就跳转到 * LM1_345 行。
  如果 M1 = 6,M1 = 7,就跳转到 * L167 行。
11 MOV P500 '—— M1 不等于 1 ~7 就跳转到本行。
100  * ABC1'——程序分支标志
101 MOV P100 '——前进到 P100 点。
```

```
102' ……
110 MOV P200 '——前进到 P200 点。
111 * LJMP'——程序分支标志
112 MOV P300 '——前进到 P300 点。
113' ……
170 * L67'——程序分支标志
171 MOV P600 '——前进到 P300 点。
172' ……
200 * LM1_345'——程序分支标志
201 MOV P400 '——前进到 P300 点。
202' ……
```

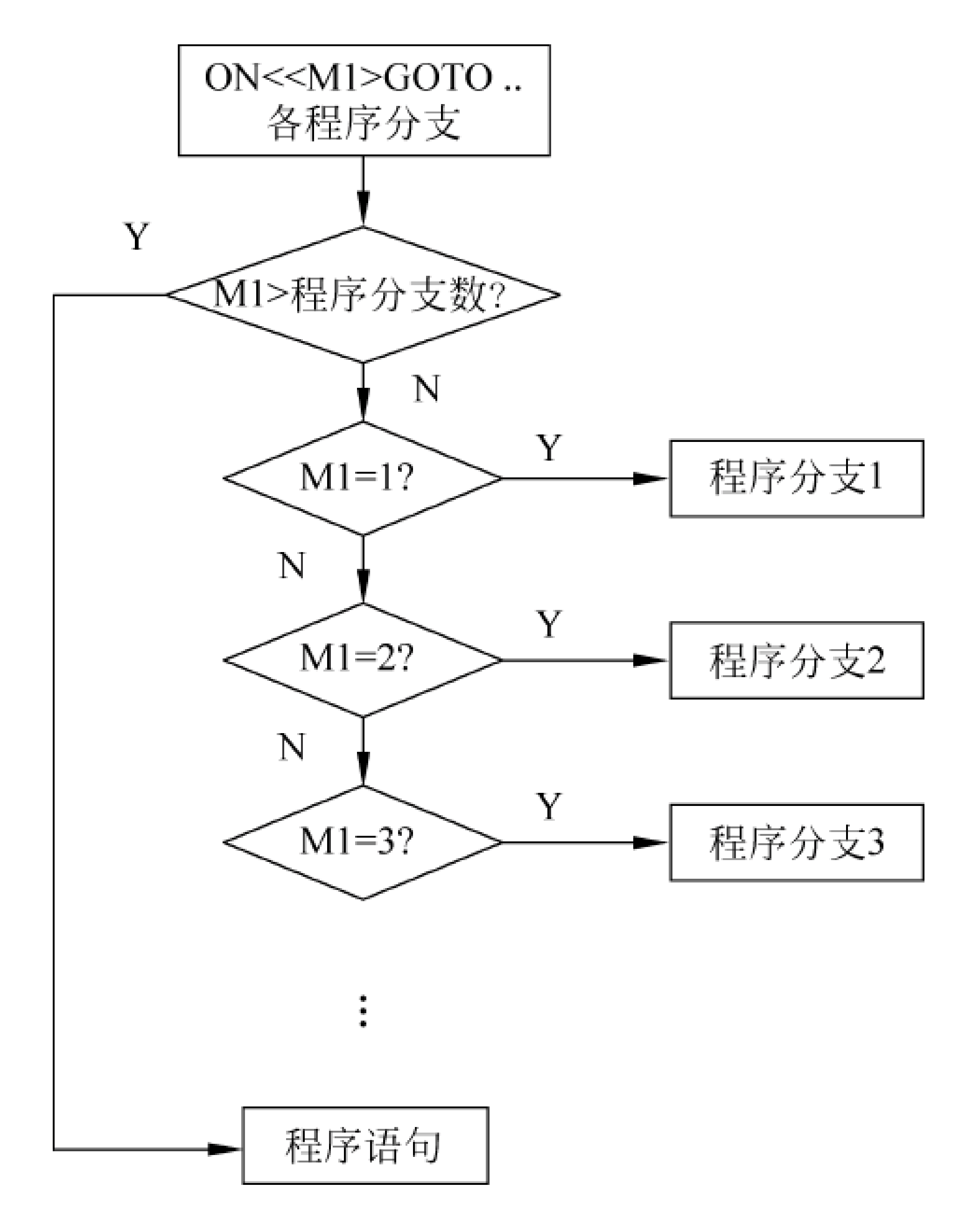


图 12-2 ON…GoTo 指令的流程

12.5 子程序指令 GoSub

1. 功能

本指令为调用子程序指令。子程序前有 * 标志。在子程序中必须要有返回指令 Return。这种调用方法与 CallP 指令的区别是：GoSub 指令指定的“子程序”写在“同一程序”内，用“标签”标定“起始行”，以 Return 作为子程序结束并返回“主程序”；而 CALLP 指令调用的程序可以是一个独立的程序。

2. 指令格式

```
GoSub <子程序标签>
```

3. 指令例句

```
10 GoSub * LBL
```

```
11  End
…
100  * LBL
101  Mov P1
102  Return '——务必写 Return 指令
```

4. 说明

(1) 子程序结束务必写 Return 指令，不能使用 GoTo 指令。

(2) 在子程序中还可使用 GoSub 指令，可以使用 800 段。

12.6　子程序调用指令 CallP

1. 功能

本指令用于调用子程序。

2. 指令格式及说明

```
CallP [程序名][自变量 1][自变量 2]
```

(1) 程序名——被调用的子程序名字。

(2) [自变量 1]、[自变量 2]——设置在子程序中使用的变量，类似于局部变量，只在被调用的子程序中有效。

3. 指令例句 1

调用子程序时同时指定自变量。

```
1  M1 = 0
2  CallP "10" ,M1,P1,P2'——调用"10"号子程序,同时指定 M1,P1,P2 为子程序中使用的变量。
3  M1 = 1
4  CallP "10" ,M1,P1,P2'——调用"10"号子程序,同时指定 M1,P1,P2 为子程序中使用的变量。
10 CallP "10", M2,P3,P4'——调用"10"号子程序,同时指定 M2,P3,P4 为子程序中使用的变量。
15 End
```

“10”子程序：

```
1  FPrm M01, P01,P02'——规定与主程序中对应的"变量"
2  If M01 <> 0 Then GoTo * LBL1'——判断语句
3  Mov P01'——前进到 P01 点
4  * LBL1'——程序分支标志
5  Mvs P02'——前进到 P02 点
6  End'——结束(返回主程序)
```

注意：在主程序第 1 步、第 4 步调用子程序时，“10”子程序变量 M01，P01，P02 与主程序指定的变量 M1，P1，P2 相对应。

在主程序第 10 步调用子程序时，“10”子程序变量 M01，P01，P02 与主程序指定的变量 M2，P3，P4 相对应。

主程序与子程序的关系如图 12-3 所示。

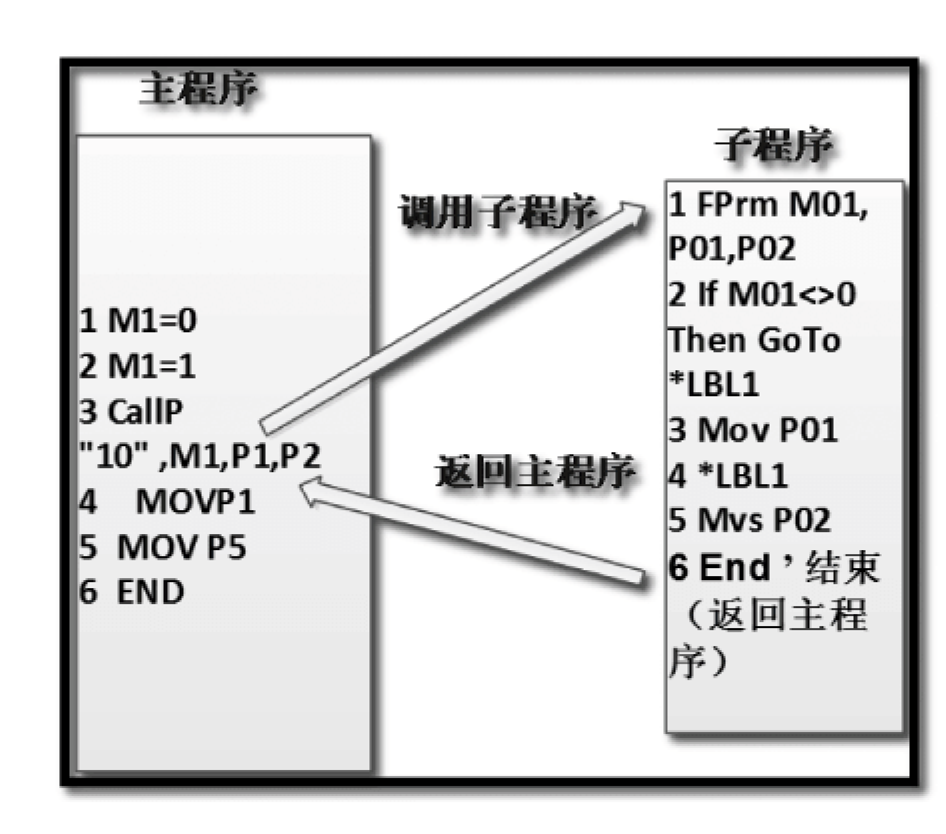


图 12-3　主程序与子程序的关系

4. 指令例句 2

调用子程序时不指定自变量：

```
Mov P1'——前进到 P1 点
CallP "20" '——调用"20"号子程序
Mov P2'——前进到 P1 点
CallP "20"'——调用"20"号子程序
End
```

“20”子程序：

```
Mov P1 ------ '子程序中的 P1 与主程序中的 P1 不同
Mvs P002'——前进到 P002 点
M_Out(17) = 1'——赋值
End'——结束
```

5. 说明

(1) 子程序以 END 结束并返回主程序。如果没有 END 指令，则在最终行返回主程序。

(2) CallP 指令指定自变量时，在子程序一侧必须用 FPrm 定义自变量，而且数量类型必须相同，否则发生报警。

(3) 可以执行 8 级子程序调用。

(4) TOOL 数据在子程序中有效。

12.7 FPrm

1. 功能

从主程序中调用子程序指令时，如果规定有自变量，就用本指令使主程序定义的“局部变量”在子程序中有效。

2. 指令格式

```
FPrm <假设自变量><假设自变量>
```

3. 指令例句

主程序：

```
M1 = 1'——赋值
P2 = P_Curr'——设置 P2 为当前点
P3 = P100'——赋值
CallP "100",M1,P2,P3'——调用子程序 "100", 同时指定了变量 M1, P2,P3
```

子程序“100”：

```
FPrm M1,P2,P3'——指令从主程序中定义的变量有效
If M1 = 1 Then GoTo *LBL'——判断执行语句
Mov P1'——前进到 P1 点
*LBL'——程序分支标志
Mvs P2'——前进到 P2 点
End'——程序结束
```

12.8 子程序调用指令 On GoSub

1. 功能

根据不同的条件调用不同的子程序，指令流程参看图 12-4。

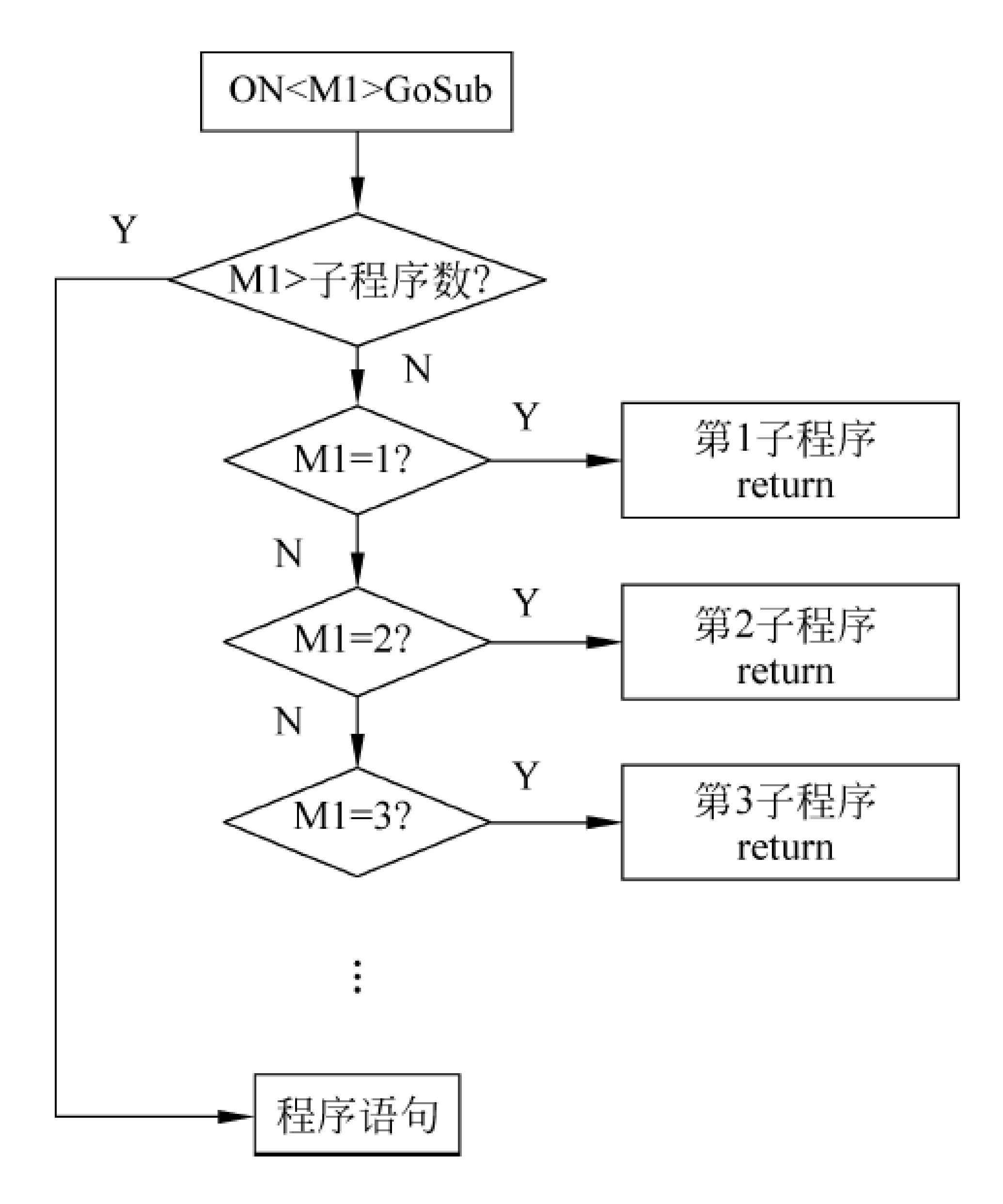


图 12-4 ON…GoSub 指令的流程

2. 格式

```
On <式> GoSub[<子程序标记> ] [, [<子程序标记>] ]
```

3. 术语

<式>——数值运算式(作为判断条件)。

<子程序标记>——记述子程序标记名，最大数为 32。

4. 指令样例

根据 M1 数值(1～7)调用不同的子程序。

```
(M1 = 1 时调用子程序 ABC1,M1 = 2 时调用子程序 Lsub、M1 = 3、4、5 时调用子程序 LM1_345,M1 = 6、7
时调用子程序 L67)
1   M1 = M_Inb(16) And &H7 '——赋值
2   On M1 GoSub * ABC1, * Lsub, * LM1_345, * LM1_345, * LM1_345, * L67, * L67 '——子程序调用指
令(注意,有 7 个子程序)
100  * ABC1 '——子程序标志
101  '——M1 = 1 时的程序处理
102 Return'——务必以 Return 返回主程序
121  * Lsub'——子程序标志
122  '——M1 = 2 时的程序处理
123 Return'——务必以 Return 返回主程序
170  * L67 '——子程序标志
171  '—— M1 = 6,M1 = 7 时的程序处理
```

```
172 Return'——务必以 Return 返回主程序
200 *LM1_345'——子程序标志
201 'M1 = 3、M1 = 4、M1 = 5 时的子程序
202 Return'——务必以 Return 返回主程序
```

5. 说明

(1) 以<数值运算式>的值决定调用某个子程序。

例如,<数值运算式>的值=2,即调用第 2 号记述的子程序。

(2) <数值运算式>的值大于<调用子程序>个数时,就跳转到下一行。例如,<数值运算式>的值=5,<调用子程序>=3 的情况下,会跳转到下一行。

(3) 子程序结束处必须写 Return,以返回主程序。

12.9 需要思考的问题

(1) 什么是程序结构?

(2) If…Then…Else…EndIf 指令在什么地方结束?中间可以有多少层?

(3) GoSub 指令与 CallP 指令有什么不同?

(4) 对于不同的条件选择不同的子程序可以使用什么指令?

(5) 对于不同的条件选择不同的程序分支可以使用什么指令?

(6) Return 指令起什么作用?

第 13 章 第 13 日——编程指令的学习和使用(4)

【学习目的】

本章继续进行编程指令的深度学习,本章学习的循环指令、中断指令也是属于程序结构方面的指令。

13.1 循 环 指 令

1. 功能

本指令为循环动作指令。如果满足循环条件,则循环执行 While…Wend 之间的动作;如果不满足循环条件则跳出循环。

2. 指令格式

```
While    <循环条件>
处理动作
WEnd
```

<循环条件>——数据表达式

3. 指令例句

如果 M1 为－5 和 5,则循环执行,程序如下。

```
While (M1 >=- 5) And (M1 <= 5)'——如果 M1 为 - 5 和 5,则循环执行
M1 =- (M1 + 1)'——循环条件处理
M_Out(8) = M1 '——赋值
WEnd'——循环结束指令
End '——结束
```

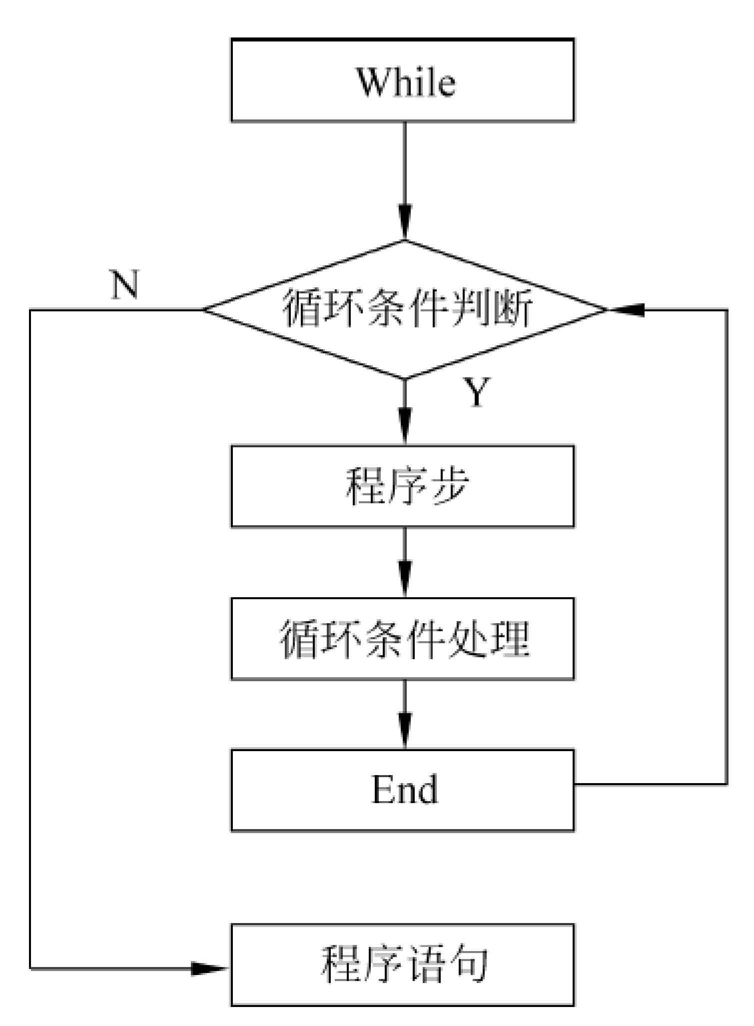


图 13-1 循环语句流程

本指令的循环过程如图 13-1 所示。

13.2 中 断

13.2.1 Def Act 中断指令

1. 功能

本指令用于定义执行中断程序的条件及中断程序的动作。

2. 指令格式及说明

Def Act <中断程序级别><条件><执行动作><类型>

(1) <中断程序级别>——设置中断程序的级别(中断程序号)。

(2) <条件>——是否执行“中断程序”的判断条件。

(3) <执行动作>——中断程序动作内容。

(4) <类型>——中断程序的执行时间点,也就是主程序的停止类型。

① 省略:停止类型1,以100%速度倍率正常停止。

② S:停止类型2,以最短时间、最短距离减速停止。

③ L:停止类型3,执行完当前程序行后才停止。

3. 指令例句

1 Def Act 1,M_In(17) = 1 GoSub *L100'——定义Act 1中断程序为:如果输入信号17 = ON,则跳转到子程序*L100。

2 Def Act 2,MFG1 And MFG2 GoTo *L200'——定义Act 2中断程序:如果MFG1与MFG2的"逻辑AND"运算=真,则跳转到子程序*L200。

3 Def Act 3,M_Timer(1)> 10500 GoSub *LBL'——定义Act 3中断程序为:如果计时器时间大于10500ms则跳转到子程序*LBL。

10 *L100:M_Timer(1) = 0'——计时器M_Timer(1)设置=0。

11 Act 3 = 1'——Act 3动作区间有效。

12 Return 0'——返回。

…

20 *L200'——程序分支标志。

21 Mov P_Safe'——前进到安全点。

22 End'——结束。

…

30 *LBL'——程序分支标志。

31 M_Timer(1) = 0'——计时器M_Timer(1)设置=0。

32 Act 3 = 0'——Act 3动作区间无效。

32 Return 0'——返回。

4. 说明

(1) 中断程序从“跳转起始行”到Return结束;

(2) 中断程序级别以号码1~8表示,数字越小越优先,如Act 1优先于Act 2;

(3) 执行中断程序时,主程序的停止类型如图13-2和图13-3所示。

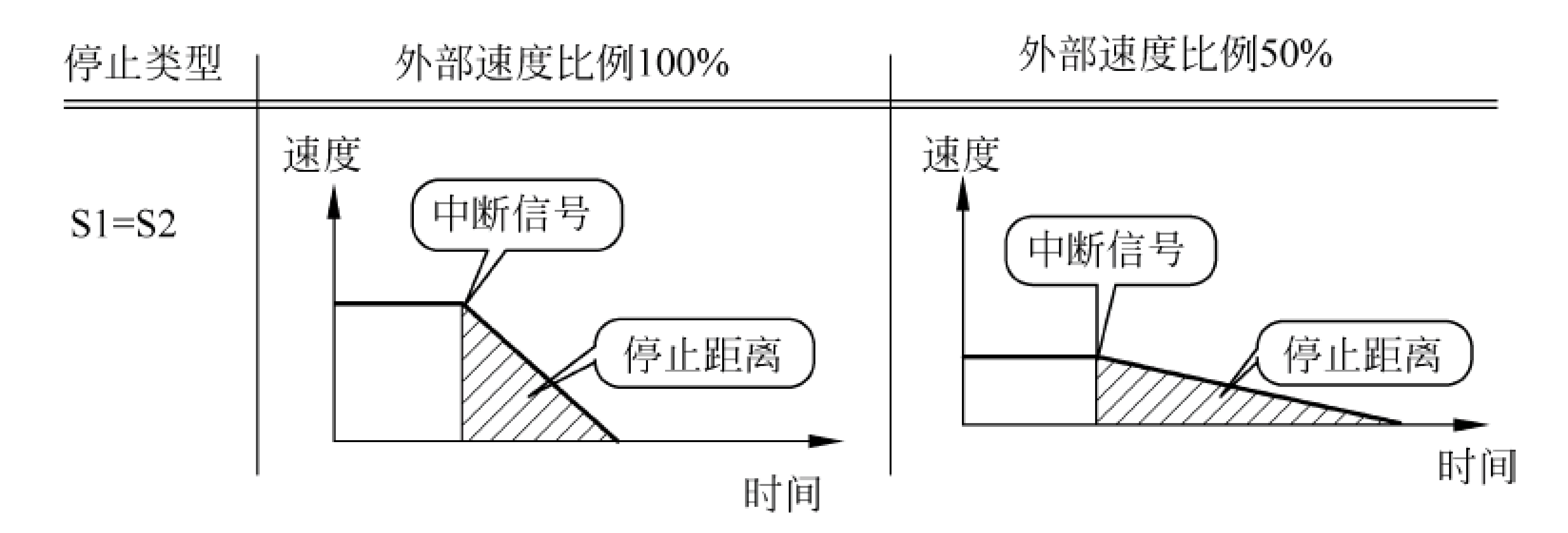


图13-2 停止类型1——停止过程中的行程相同

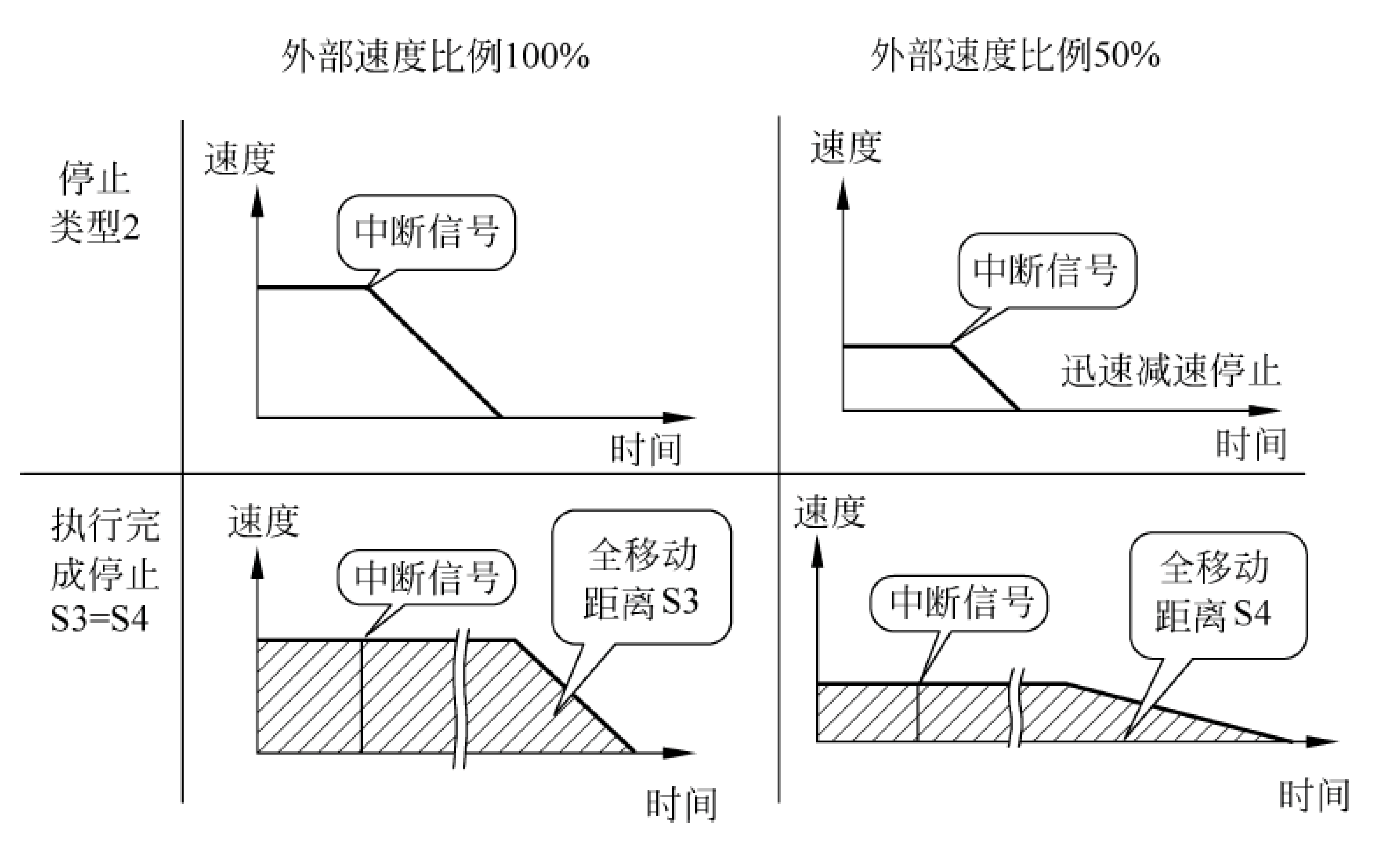


图 13-3 停止类型 2——以最短时间,最短距离减速停止

13.2.2 Act 指令

1. 功能

ACT 指令有以下两重意义。

ACT 1～ACT 8 是“中断程序”的程序级别标志。

ACT n=1,ACT n=0 划出了中断程序 ACT n 的生效区间。

2. 指令格式

```
ACT  <被定义的程序级别标志> = < 1 >——中断程序可执行区间起始标志
ACT  <被定义的程序级别标志> = < 0 >——中断程序可执行区间结束标志
```

指令格式说明:

<被定义的程序级别标志>——设置中断程序的“程序级别标志”。

3. 指令例句 1

```
Def Act 1,M_In(1) = 1 GoSub * INTR'——定义 Act 1 对应的"中断程序"。
Mov P1'——前进到 P1 点。
Act 1 = 1'——"Act 1 定义的中断程序"动作区间生效。
Mov P2'——前进到 P2 点。
Act 1 = 0'——"Act 1 定义的中断程序"动作区间无效。
 * INTR'——程序分支标志
If M_In(1) = 1 GoTo * INTR'——判断执行语句
Return 0'——返回
```

4. 指令例句 2

```
Def Act 1,M_In(1) = 1 GoSub * INTR'——定义 Act 1 对应的"中断程序"
Mov P1'——前进到 P1 点。
Act 1 = 1'——Act 1 动作区间生效
Mov P2'——前进到 P2 点。
 * INTR'——程序分支标志
```

```
6  Act 1 = 0'——Act 1 动作区间无效
7  M_Out(10) = 1'——赋值
    Return 1'——结束
```


5. 说明

(1) Act 0 为最优先状态。程序启动时即为“Act 0＝1”状态。如果“Act 0＝0”,则“Act 1～8＝1”也无效。

(2) 中断程序的结束(返回)由“Return 1”或“Return 0”指定。

Return 1 ——转入主程序的下一行;

Return 0 ——跳转到主程序中“中断程序”的发生行。

13.3　暂停指令 HLT

1. 功能

本指令为暂停执行程序,程序处于待机状态。如果发出再启动信号,从程序的下一行启动。本指令在分段调试程序时常用。

2. 指令格式

```
Hlt
```

3. 指令例句 1

```
1 Hlt'——无条件暂停执行程序。
```

4. 指令例句 2

满足某一条件时,执行暂停。

```
100  If M_In(18) = 1 Then Hlt'——如果输入信号 18 = ON,则暂停。
200  Mov P1 WthIf M_In(17) = 1, Hlt '——在向 P1 点移动过程中,如果输入信号 17 = ON,则暂停。
```

5. 说明

(1) 在 Hlt 暂停后,重新发出启动信号,程序从下一行启动执行。

(2) 如果是在附随语句中发生的暂停,重新发出启动信号后,程序从中断处启动执行。

13.4　暂停指令 DLY

1. 功能

本指令用于设置程序中的“暂停时间”,也作为构成“脉冲型输出”的方法。

2. 指令格式

(1) 程序暂停型

```
Dly<暂停时间>
```

(2) 设定输出信号＝ON 的时间(构成脉冲输出)

```
M_Out(1) = 1 Dly<时间>
```

3. 指令例句 1

```
Dly 30 '——程序暂停时间 30s。
```

4. 指令例句 2

设定输出信号＝ON 的时间(构成脉冲输出)：

```
M_Out(17) = 1 Dly 0.5'——输出端子(17) = ON 时间为 0.5s
M_Outb(18) = 1 Dly 0.5'——输出端子(18) = ON 时间为 0.5s
```

13.5 需要思考的问题

(1) 循环指令也是改变程序结构的指令吗？

(2) 如何制作循环指令的循环条件？

(3) 中断指令主要用于什么场合？

(4) 中断指令分级吗？

(5) 如何定义中断指令的有效工作区间？如果在有效工作区间之外，某中断指令的条件满足，程序会如何运行？

(6) 中断指令执行完毕后会返回主程序吗？

(7) DLY 指令与 HLT 指令有什么区别？在需要暂时停止执行程序时用哪一种指令？

第 14 章　第 14 日——状态变量的学习和使用

【学习目的】

机器人的工作状态如“当前位置”等是可以用变量的形式表示的。实际上，每一种工业控制器都有表示自身工作状态的功能，如数控系统用“X 接口”表示工作状态，所以机器人的状态变量就是表示机器人的“工作状态”的数据，在实际应用中极为重要。本章将详细解释各机器人状态变量的定义、功能和使用方法。

14.1　常用状态变量 P_Curr——当前位置（X，Y，Z，A，B，C，L1，L2）（FL1，FL2）

1. 功能

P_Curr 为“当前位置”，这是最常用的变量。

2. 格式

```
<位置变量> = P_Curr <机器人编号>
<位置变量>——以 P 开头，表示"位置点"的变量
<机器人编号>——1～3，省略时为 1
```

3. 例句

```
1   Def Act 1,M_In(10) = 1 GoTo ＊LACT'——定义一个中断程序。
2   Act 1 = 1'——中断区间有效。
3   Mov P1'——前进到 P1 点。
4   Mov P2'——前进到 P2 点。
5   Act 1 = 0'——中断区间无效。
100 ＊LACT'——程序分支标志。
101 P100 = P_Curr'——读取当前位置。设置 P100 = 当前位置。
102 Mov P100, - 100'——移动到 P100 近点 - 100 的位置。
103 End'——结束。
```

14.2　P_Fbc——以伺服反馈脉冲表示的当前位置（X，Y，Z，A，B，C，L1，L2）（FL1，FL2）

1. 功能

P_Fbc 是以伺服反馈脉冲表示的当前位置（X，Y，Z，A，B，C，L1，L2）（FL1，FL2）。

2. 格式

```
<位置变量> = P_Fbc  <机器人编号>
<机器人编号>——1~3,省略时为 1
```

3. 例句

```
1  P1 = P_Fbc' ——P1 点为以脉冲表示的当前位置。
```

14.3 常用状态变量 J_Curr——各关节轴的当前位置数据

1. 功能

J_Curr 是以各关节轴的旋转角度表示的“当前位置”数据，在编写程序时是经常使用的重要数据。

2. 格式

```
<关节型变量> = J_Curr <机器人编号>
```

<关节型变量>：注意要使用“关节型的位置变量”——以 J 开头。

<机器人编号>：设置范围 1～3。

3. 例句

```
J1 = J_Curr' ——设置 J1 为关节型当前位置点
```

14.4 J_ECurr——当前编码器脉冲数

1. 功能

J_ECurr 为各轴编码器发出的“脉冲数”。

2. 格式

```
<关节型变量> = J_ECurr <机器人编号><关节型变量>——注意要使用"关节型的位置变量",以 J
开头。
<机器人编号>——设置范围 1~3。
```

3. 例句

```
1  JA = J_ECurr(1)'——设置 JA 为各轴脉冲值
2  MA = JA.J1' ——设置 MA 为 J1 轴脉冲值
```

14.5 常用状态变量 J_Fbc/J_AmpFbc——关节轴的当前位置/关节轴的当前电流值

1. 功能

(1) J_Fbc——以编码器实际反馈脉冲表示的关节轴当前位置。

(2) J_AmpFbc——关节轴的当前电流值。

2. 格式

```
(1) <关节型变量> = J_Fbc <机器人编号>
(2) <关节型变量> = J_AmpFbc <机器人编号>
```

<关节型变量>：注意要使用“关节型的位置变量”，以J开头。

<机器人编号>：设置范围1～3。

3. 例句

```
1  J1 = J_Fbc'——J1 = 以编码器实际反馈脉冲表示的关节轴当前位置。
2  J2 = J_AmpFbc'——J2 = 各轴当前电流值。
```

14.6 M_In/M_Inb/M_In8/M_Inw/M_In16 ——输入信号状态

1. 功能

这是一类输入信号状态，是最常用的状态信号。

M_In——位信号。

M_Inb/M_In8——以“字节”为单位的输入信号。

M_Inw/M_In16——以“字”为单位的输入信号。

2. 格式

```
(1) <数值变量> = M_In <数式>
(2) <数值变量> = M_Inb <数式>或 M_In8 <数式>
(3) <数值变量> = M_Inw <数式>或 M_In16 <数式>
```

3. 说明

<数式>——输入信号地址。输入信号地址的分配定义如下。

(1) 0～255：通用输入信号。

(2) 716～731：多抓手信号。

(3) 900～907：抓手输入信号。

(4) 2000～5071：PROFIBUS用。

(5) 6000～8047：CC-Link用。

4. 例句

```
1  M1% = M_In(10010)'—— M1 = 输入信号 10010 的值(1 或 0)。
2  M2% = M_Inb(900)'—— M2 = 输入信号 900～907 的 8 位数值。
3  M3% = M_Inb(10300) And &H7'—— M3 = 10300～10307 与 H7 的逻辑和运算值。
4  M4% = M_Inw(15000)'—— M4 = 输入 15000～15015 构成的数据值(相当于一个 16 位的数据寄存器)。
```

14.7 M_Out/M_Outb/M_Out8/M_Outw/M_Out16 ——输出信号状态

1. 功能

输出信号状态。

(1) M_Out——以“位”为单位的输出信号状态。

(2) M_Outb/M_Out8——以“字节(8 位)”为单位的输出信号数据。

(3) M_Outw/M_Out16——以“字(16 位)”为单位的输出信号数据。

这是最常用的变量之一。

2. 格式

```
(1) M_Out(<数式 1 >) = <数值 2 >
(2) M_Outb(<数式 1 >)或 M_Out8(<数式 1 >)  = <数值 3 >
(3) M_Outw(<数式 1 >)或 M_Out16(<数式 1 >)  = <数值 4 >
(4) M_Out(<数式 1 >) = <数值 2 > dly <时间>
<数值变量> = M_Out(<数式 1 >)
```

3. 说明

<数式 1 >——用于指定输出信号的地址。输出信号的地址分配如下。

(1) 10000～18191：多 CPU 共用软元件。

(2) 0～255：外部 I/O 信号。

(3) 716～723：多抓手信号。

(4) 900～907：抓手信号。

(5) 2000～5071：PROFIBUS 用信号。

(6) 6000～8047：CC-Link 用信号。

(7) <数值 2 >,<数值 3 >,<数值 4 >：输出信号输出值，可以是常数、变量、数值表达式。

(8) <数值 2 >设置范围：0 或 1。

(9) <数值 3 >设置范围：－128 ～＋127。

(10) <数值 4 >设置范围：－32 768＋32 767。

(11) <时间>：设置输出信号＝ON 的时间。单位：秒。

4. 例句

```
1  M_Out(902) = 1 '——指令输出信号 902 = ON。
2  M_Outb(10016) = &HFF'——指令输出信号 10016 ～10023 的 8 位 = ON。
3  M_Outw(10032) = &HFFFF '——指令输出信号 10032 ～10047 的 16 位 = ON。
4  M4 = M_Outb(10200) And &H0F'—— M4 = (输出信号 10200～10207)与 H0F 的逻辑和。
```

5. 说明

输出信号与其他状态变量不同。输出信号是可以对其进行“指令”的变量而不仅仅是“读取其状态”的变量。实际上更多的是对输出信号进行设置，指令输出信号＝ON/OFF。

14.8 需要思考的问题

(1) 为什么要了解机器人的工作状态？

(2) 什么是机器人的“当前位置”？机器人的“当前位置”有几种表示方法？

(3) 输入信号有几种表示方式？

(4) 输出信号有几种表示方式？

(5) 输出信号仅仅是只读信号吗？

(6) 输入输出的地址分配有什么意义？

第 15 章　第 15 日——机器人系统的特殊功能

【学习目的】

机器人不同于一般的运动机械，机器人具备多轴（6 轴）联动功能，其应用范围也有其特殊性，所以机器人有许多其他运动机械所没有的功能，本章将对这些特殊功能进行介绍，以便读者在实际应用时使用这些功能。

15.1　操　作　权

1. 能够对机器人进行控制的设备

对机器人进行控制的设备有以下几种。

（1）示教单元；

（2）操作面板（外部信号）；

（3）计算机；

（4）触摸屏。

某一类设备对“机器人”的控制权就称为“操作权”。示教单元上有一个“使能开关”就是“操作权”开关。如表 15-1 所示是示教单元上的“使能开关”与“操作权”的关系。

表 15-1　示教单元上的“使能开关”与“操作权”的关系

设定开关	使能开关	无效		有效	
	控制器	自动	手动	自动	手动
操作权	示教单元	NO	NO	NO	YES
	控制器操作面板	YES	NO	NO	NO
	计算机	YES	NO	NO	NO
	外部信号	YES	NO	NO	NO

YES——有操作权，NO——无操作权。

2. 与操作权相关的参数

IOENA——本信号的功能是使外部操作信号有效和无效。在 RT TooLBox2 软件中的“参数”→“通用 1”中设置本参数。

操作权：对机器人的操作可能来自①示教单元；②外部信号；③计算机软件（调试时）；④触摸屏。

3. 实际操作

实际操作如下。

(1) 在示教单元中 ENABLE 开关＝ON，可以进行示教操作。即使外部 IO 操作权＝ON，即使外部没有选择"自动模式"，也可以通过示教单元的"开机"→"运行"→"操作"面板→"启动"进行程序"启动"(示教单元有优先功能 ENABLE)。

(2) 如果在操作面板上选择了"自动模式"，而 ENABLE＝ON，系统会报警，使 ENABLE＝OFF，报警消除。

(3) 如果需要进入调试状态，必须使 IOENA＝OFF。

(4) 如果使用外部信号操作，则需要使 IOENA＝ON。

15.2 其他功能

1. 最佳速度控制

最佳速度控制功能是指机器人在两点之间运动，需要在保持形位要求的同时，还需要控制速度，防止速度过大出现报警。

最佳速度控制功能有效时，机器人控制点速度不固定。用 Spd M_NSpd 指令可设置"最佳速度控制"。

2. 最佳加减速度控制

最佳加减速度控制是指机器人根据加减速时间、抓手及工件重量、工件重心位置，自动设置最佳加减速时间的功能。

用 Oadl(Optimal Acceleration)指令设置最佳加减速度控制。

3. 柔性控制功能

柔性控制功能是指对机器人的综合力度进行控制的功能。通常用于压嵌工件的动作。(以直角坐标系为基础)根据伺服编码器反馈脉冲，进行机器人柔性控制。用 Cmp Too 指令设置"伺服柔性控制功能"。

4. 碰撞检测功能

碰撞检测功能是指在自动运行和 JOG 运行中，系统时刻检测 TOOL 或机械臂与周边设备的碰撞干涉状态。用 ColChk (Col Check)指令设置"碰撞检测功能"的有效/无效。

机器人配置有对"碰撞而产生的异常"进行检测的"碰撞检测功能"，出厂时将"碰撞检测功能"设置为无效状态。"碰撞检测功能"的有效/无效状态切换可通过参数 COL 及 ColChk 指令完成，必须作为对机器人及外围装置的保护加以运用。

"碰撞检测功能"是通过机器人的动力学模型，在随时推算动作所需的扭矩的同时，对异常现象进行检测的功能。因此，当抓手、工件条件的设置(参数：HNDDAT＊、WRKDAT＊的设置值)与实际相差过大时，或是速度、电机扭矩有急剧变动的动作(特殊点附近的直线动作或反转动作，低温状态或长期停止后启动运行)，急剧的扭矩变动就会被检测为"碰撞"。简单地说，就是一直检测"计算转矩与实际转矩的差值"，当该值过大时，就报警。

5. 连续轨迹控制功能

在多点连续定位时，使运动轨迹为一连续轨迹。本功能可以避免多次的分段加减速从而提高效率。用 Cnt(Continuous)指令设置"连续轨迹控制功能"。

6. 附加轴控制

控制行走台等外部伺服驱动系统。外部伺服轴相对于机器人而言即为"附加轴"。

7. 多机器控制

可控制多台机器人。

8. 与外部机器通信功能

机器人与外部机器通信功能有下列方法。

(1) 通过外部 I/O 信号:

CR750Q——PLC 通信,输入 8192/输出 8192。

CR750D——输入 256/输出 256。

(2) 与外部数据的链路通信。

所谓数据链路指与外部机器(视觉传感器等)收发补偿量等数据,通过"以太网端口"进行。

9. 码垛指令功能

机器人配置有码垛指令,有多行、单行、圆弧码垛指令。实际上是确定矩阵点格中心点位置的指令。

10. 用户定义区

用户可设置 32 个任意空间,监视机器人前端控制点是否进入该区域,将机器人状态输出到外部并报警。

可以用下列三种方法限制机器人动作范围。

(1) 关节轴动作范围限制(J1~J6)。

(2) 以直角坐标系设置限制范围。

(3) 以任意设置的平面为界面设置限制范围(在平面的前面或后面),由参数 SFCnAT 设置。

15.3 需要思考的问题

(1) 什么是操作权?

(2) 什么是碰撞检测功能?

(3) 什么是柔性控制功能?

(4) 什么是最佳速度控制功能?

第 16 章　第 16 日——参数的功能及设置(1)

【学习目的】

参数用于表示机器人的不同工作性能。设置不同的参数可以给机器人赋予不同的性能。本章要学习机器人的动作型参数和程序型参数。

16.1　动作型参数

16.1.1　动作型参数一览表

动作型参数一览表如表 16-1 所示。

表 16-1　动作型参数一览表

序号	参数类型	参数符号	参数名称	参数功能
1	动作	MEJAR	动作范围	用于设置各关节轴旋转范围
2	动作	MEPAR	各轴在直角坐标系行程范围	设置各轴在直角坐标系内的行程范围
3	动作	Useprog	用户设置的原点	用户自行设置的原点
4	动作	MELTEXS	机械手前端行程限制	用于限制机械手前端对基座的干涉
5	动作	JOGJSP	JOG 步进行程和速度倍率	设置关节轴的 JOG 的步进行程和速度倍率
6	动作	JOGPSP	JOG 步进行程和速度倍率	设置以直角坐标系表示的 JOG 的步进行程和速度倍率
7	动作	MEXBS	基本坐标系偏置	设置“基本坐标系原点”在“世界坐标系”中的位置(偏置)
8	动作	MEXTL	标准工具坐标系“偏置”(TOOL 坐标系也称为抓手坐标系)	设置“抓手坐标系原点”在“机械 IF 坐标系”中的位置(偏置)
9	动作	MEXBSNO	世界坐标系编号	设置世界坐标系编号
10	动作	AREA * AT	报警类型	设置报警类型
	动作	USRAREA	报警输出信号	设置输出信号
11	动作	AREASP *	空间的一个对角点	设置“用户定义区”的一个对角点
12	动作	AREA * CS	基准坐标系	设置“用户定义区”的“基准坐标系”
13	动作	AREA * ME	机器人编号	设置机器人“编号”
14	动作	SFC * AT	平面限制区有效/无效选择	设置平面限制区有效无效
15	动作	SFC * P1 SFC * P2 SFC * P3	构成平面的三点	设置构成平面的三点

续表

序号	参数类型	参数符号	参数名称	参数功能
16	动作	SFC＊ME	机器人编号	设置机器人"编号"
17	动作	JSAFE	退避点	设置一个应对紧急状态的退避点
18	动作	MORG	机械限位器基准点	设置机械限位器原点
19	动作	MESNGLSW	接近特异点 是否报警	设置接近特异点是否报警
20	动作	JOGSPMX	示教模式下JOG速度限制值	设置示教模式下JOG速度限制值
21	动作	WKnCORD n：1～8	工件坐标系	设置工件坐标系
22	动作	WKnWO	工件坐标系原点	
23	动作	WKnWX	工件坐标系X轴位置点	
24	动作	WKnWY	工件坐标系Y轴位置点	
25	动作	RETPATH	程序中断执行JOG动作后的返回形式	设置程序中断执行JOG动作后的返回形式
26	动作	MEGDIR	重力在各轴方向上的投影值	设置重力在各轴方向上的投影值
27	动作	ACCMODE	最佳加减速模式	设置上电后是否选择最佳加减速模式
28	动作	JADL	最佳加减速倍率	设置最佳加减速倍率
29	动作	CMPERR	伺服柔性控制报警选择	设置伺服柔性控制报警选择
30	动作	COL	碰撞检测	设置碰撞检测功能
31	动作	COLLVL	碰撞检测级别	1～5
32	动作	COLLVLJG	JOG运行时的碰撞检测级别	1～5
33	动作	WUPENA	预热运行模式	
34	动作	WUPAXIS	预热运行对象轴	
		设置	bit ON 对象轴 bit OFF 非对象轴	
35	动作	WUPTIME	预热运行时间	
		设置	单位：分(1～60)	
36	动作	WUPOVRD	预热运行速度倍率	
37	动作	HIOTYPE	抓手用电磁阀输入信号源型/漏型选择	
38	动作	HANDTYPE	设置电磁阀单线圈/双线圈及对应的外部信号	

16.1.2 动作参数详解

为了使读者更清楚参数的意义和设置，本章结合RT ToolBox软件的使用进一步解释各参数的功能，如表16-2～表16-35所示。

表16-2　MEJAR

类型	参数符号	参数名称	功　　能
动作	MEJAR	动作范围	用于设置各轴行程范围 (关节轴旋转范围)
参见图16-1			

表 16-3　MEPAR

类型	参数符号	参数名称	功　　能
动作	MEPAR	各轴在直角坐标系的行程范围	设置各轴在直角坐标系内的行程范围
参见图 16-1			

表 16-4　用户设置的原点 Useprog

类型	参数符号	参数名称	功　　能
动作	Useprog	用户设置的原点	用户自行设置的原点
用户设置的关节轴原点。以初始原点为基准，参见图 16-1			



图 16-1　行程范围及原点的设置

表 16-5　MELTEXS

类型	参数符号	参数名称	功　　能
动作	MELTEXS	机械手前端行程限制	用于限制机械手前端对基座的干涉
设置	MELTEXS＝0，限制无效；MELTEXS＝1，限制有效		

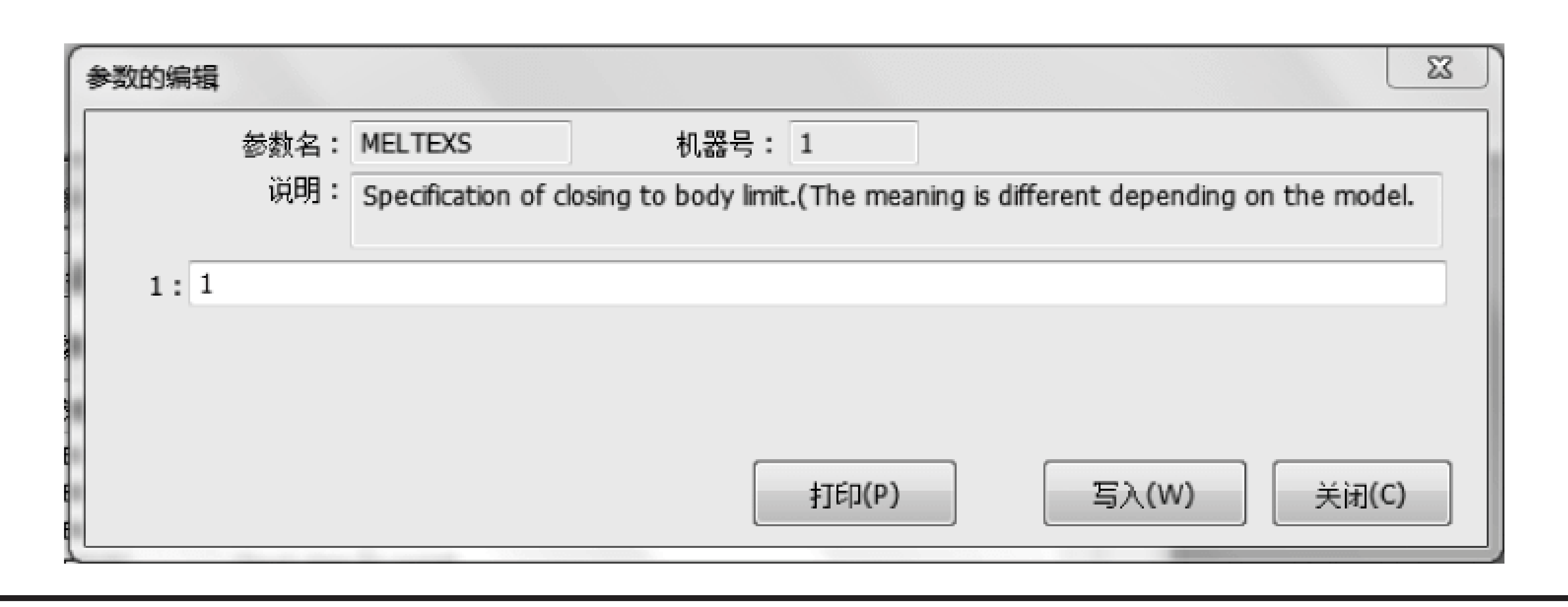


表 16-6　JOGJSP

类型	参数符号	参数名称	功　　能
动作	JOGJSP	JOG 步进行程和速度倍率	设置关节轴的 JOG 的步进行程和速度倍率
在 JOG 模式下，每按一次 JOG 按键，(轴)移动一个“定长距离”，就称为步进。参见图 16-2			

表 16-7　JOGPSP

类型	参数符号	参数名称	功　　能
动作	JOGPSP	JOG 步进行程和速度倍率	设置以直角坐标系表示的 JOG 的步进行程和速度倍率
参数 JOGPSP 与 JOGJSP 可用于示教时的精确动作，步进行程越小，调整越精确，参见图 16-2			

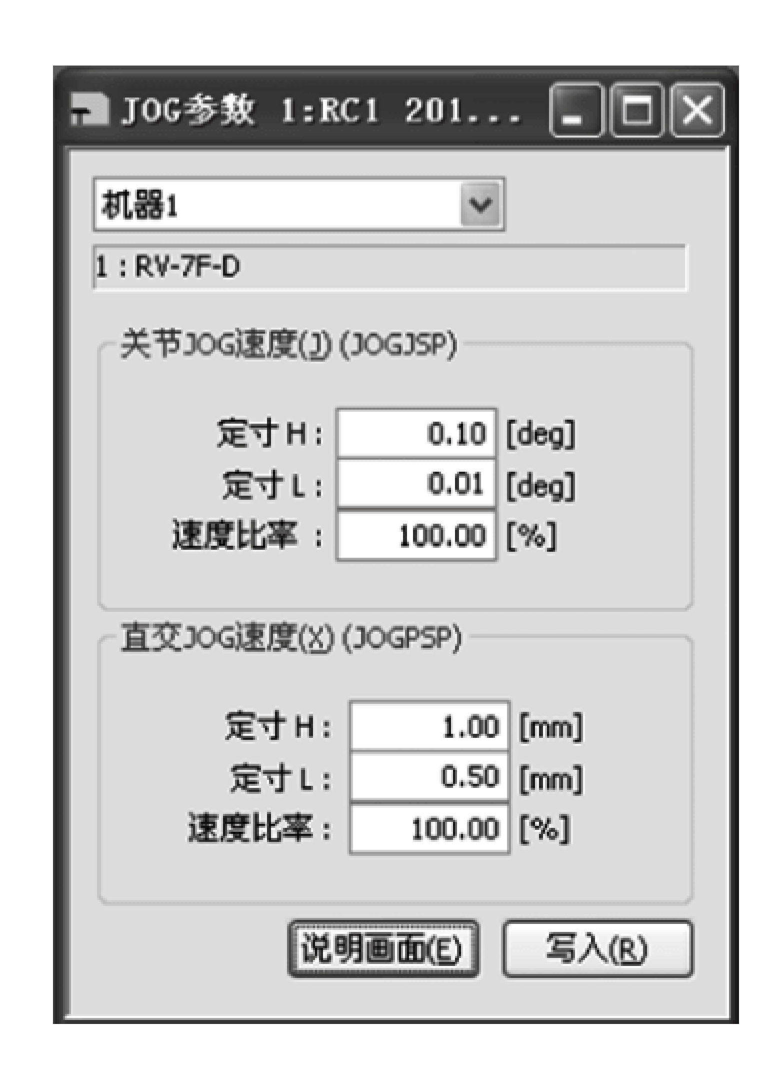


图 16-2　参数 JOGPSP 与 JOGJSP 的设置

表 16-8　MEXBS

类型	参数符号	参数名称	功　　能
动作	MEXBS	基本坐标系偏置	设置“基本坐标系原点”在“世界坐标系”中的位置(偏置)
设置	参见图 16-3		

表 16-9　MEXTL

类型	参数符号	参数名称	功　　能
动作	MEXTL	标准工具坐标系“偏置”(TOOL 坐标系也称为抓手坐标系)	设置“抓手坐标系原点”在“机械 IF 坐标系”中的位置(偏置)
设置	参见图 16-3		

表 16-10　工具坐标系“偏置”(16 个)

类型	参数符号	参数名称	功　　能
动作	MEXTL1～16	TOOL 坐标系偏置	设置 TOOL 坐标系。可设置 16 个，互相切换
参见图 16-3			

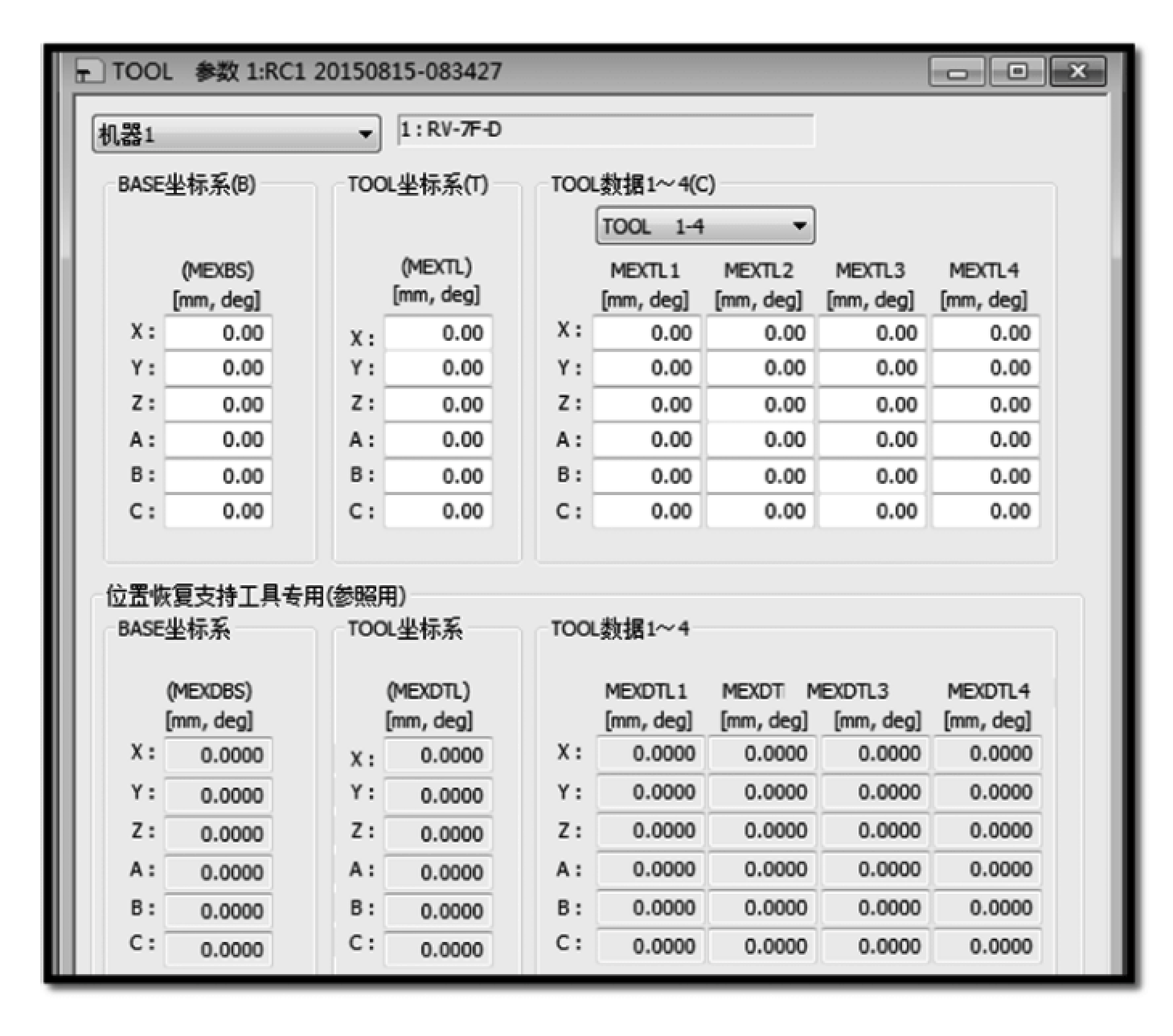


图 16-3　基本坐标系偏置和 TOOL 坐标系偏置的设置

表 16-11　世界坐标系编号

类型	参数符号	参数名称	功　　能
动作	MEXBSNO	世界坐标系编号	设置世界坐标系编号
设置	MEXBSNO＝0，初始设置；MEXBSNO＝1～8，工件坐标系。 如果是由 base 指令设置“世界坐标系”或直接设置为“标准世界坐标系”时，在读取状态下，MEXBSNO＝－1		
这样“工件坐标系”也可以理解为“世界坐标系”			

用户定义区是用户自行设定的“空间区域”。如果机器人控制点进入设定的“区域后”，系统会做相关动作。

设置方法：以两个对角点设置一个空间区域，如图 16-4 所示。

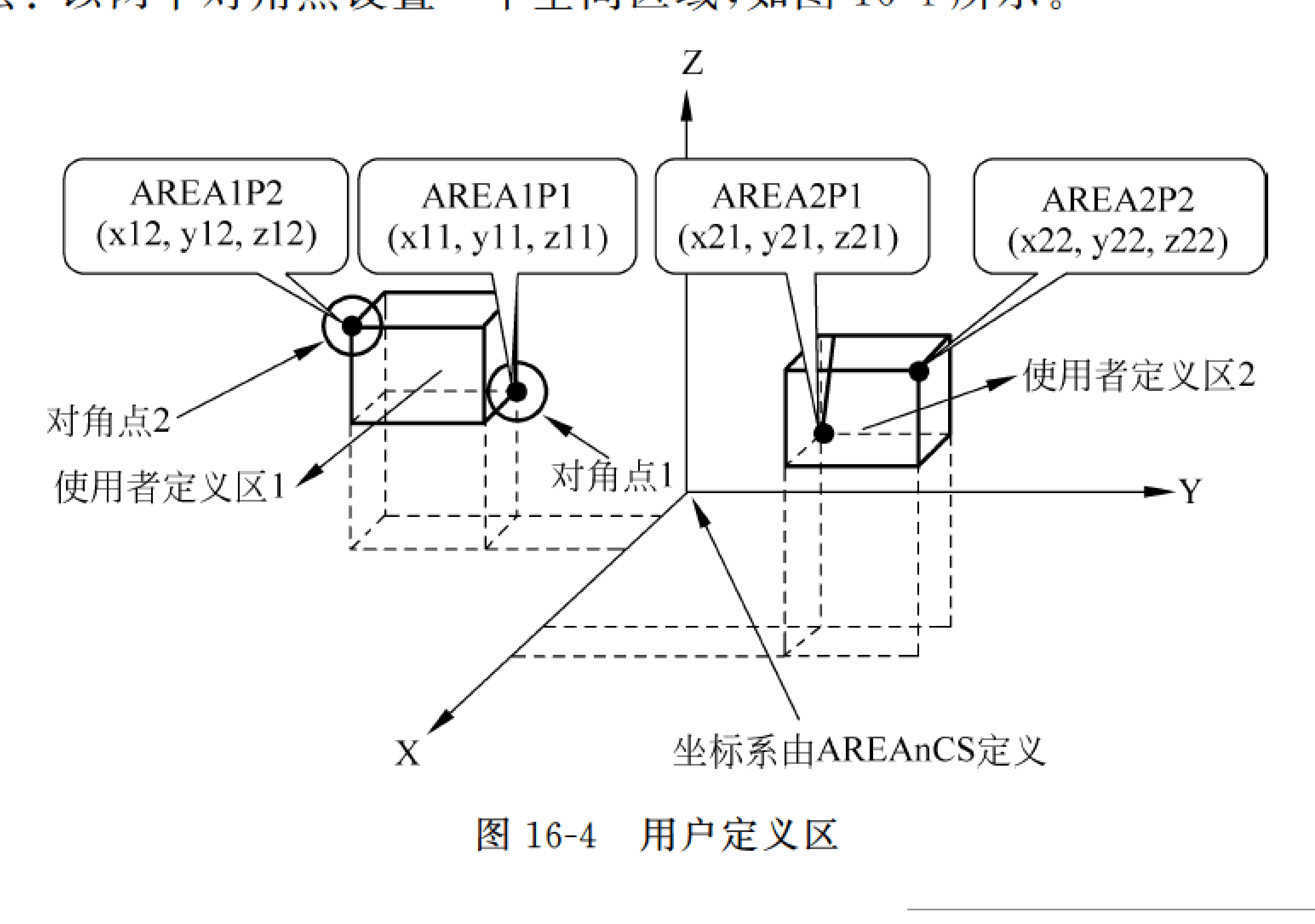


图 16-4　用户定义区

设置动作方法：如果机器人控制点进入设定的区域后，系统如何动作？可设置为：无动作/有输出信号/有报警输出。

0：无动作。

1：输出专用信号。

进入区域1，*** 信号＝ON。

进入区域2，*** 信号＝ON。

进入区域3，*** 信号＝ON。

表 16-12　AREA * AT

类型	参数符号	参数名称	功　　能
动作	AREA * AT	报警类型	设置报警类型
设置	AREA * AT＝0，无报警；AREA * AT＝1，信号输出；AREA * AT＝2，报警输出		
如图 16-5 所示			

表 16-13　USRAREA

类型	参数符号	参数名称	功　　能
动作	USRAREA	报警输出信号	设置输出信号
设置	设置最低位和最高位的输出信号（如 27～30）		
如图 16-5 所示			

表 16-14　AREASP *

类型	参数符号	参数名称	功　　能
动作	AREASP *	空间的一个对角点	设置“用户定义区”的一个对角点
设置			
如图 16-5 所示			



图 16-5　用户定义区的参数设置

表 16-15　AREA * CS

类型	参数符号	参数名称	功　　能
动作	AREA * CS	基准坐标系	设置“用户定义区”的“基准坐标系”
设置	AREA * CS=0,世界坐标系；AREA * CS=1,基本坐标系		
本参数用于选择设置“用户定义区”的“坐标系”。可以选择“世界坐标系”、“基本坐标系”			

表 16-16　AREA * ME

类型	参数符号	参数名称	功　　能
动作	AREA * ME	机器人编号	设置机器人“编号”
设置	AREA * ME=0,无效；AREA * ME=1,机器人 1(常设)；AREA * ME=2,机器人 2；AREA * ME=3,机器人 3		

自由平面限制 SFCNP1 是设置行程范围的一种方法。以任意设置的平面为界设置限制范围(在平面的前面或后面),如图 16-6 所示,由参数 SFCnAT 设置。

由三点构成一个任意平面。以这个任意平面为界限,限制机器人的动作范围,可以设置 8 个任意平面。可以规定机器人的动作范围是在原点一侧还是不在原点一侧,如图 16-6 所示。

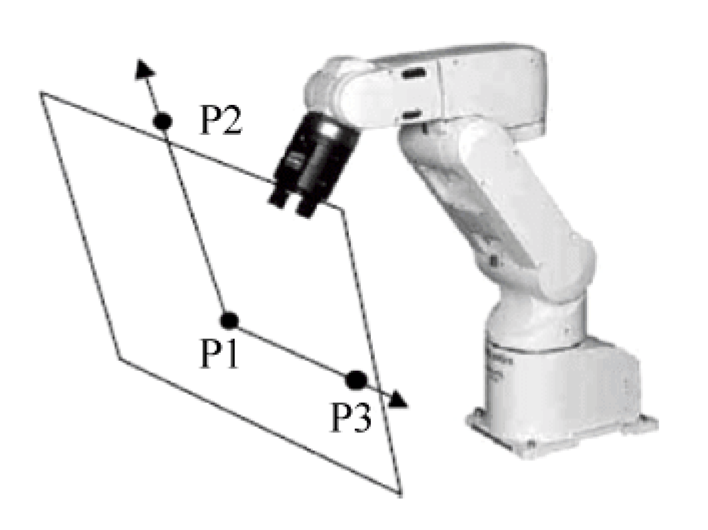


图 16-6　自由平面限制的定义

表 16-17　SFC * AT

类型	参数符号	参数名称	功　　能
动作	SFC * AT	平面限制区有效/无效选择	设置平面限制区有效/无效
设置	SFC * AT=0,无效；SFC * AT=1,可动作区在原点一侧。SFC * AT=-1,可动作区在无原点一侧		
参见图 16-7			

表 16-18　SFC * P1

类型	参数符号	参数名称	功　　能
动作	SFC * P1 SFC * P2 SFC * P3	构成平面的三点	设置构成平面的三点
设置	参见图 16-7		

表 16-19　SFC * ME

类型	参数符号	参数名称	功　　能
动作	SFC * ME	机器人编号	设置机器人“编号”
设置	SFC * ME=1,机器人 1；SFC * ME=2,机器人 2；SFC * ME=3,机器人 3		
参见图 16-7			



图 16-7　自由平面限制的参数设置

表 16-20　退避点

类型	参数符号	参数名称	功　　能
动作	JSAFE	退避点	设置一个应对紧急状态的退避点
设置	以关节轴的“度数”(deg)为单位进行设置		
参见图 16-8			

操作时,可用示教单元定好“退避点”位置。如果通过外部信号操作,则必须分配好“退避点启动”信号,如图 16-9 所示。输入信号 23 为“退避点启动”信号。



图 16-8　退避点的设置



图 16-9　启动回“退避点”信号

具体操作步骤如下。

(1) 选择自动状态;

(2) 伺服＝ON;

(3) 启动“回退避点”信号。

表 16-21　MORG

类型	参数符号	参数名称	功　　能
动作	MORG	机械限位器	设置机械限位器原点
设置	(J1,J2,J3,J4,J5,J6,J7,J8)		



表 16-22　MESNGLSW

类型	参数符号	参数名称	功　　能
动作	MESNGLSW	接近特异点是否报警	设置接近特异点是否报警
设置	MESNGLSW＝0,无效；MESNGLSW＝1,有效		



表 16-23　示教模式下 JOG 速度限制值 JOGSPMX

类型	参数符号	参数名称	功　　能
动作	JOGSPMX	示教模式下 JOG 速度限制值	设置示教模式下 JOG 速度限制值
设置			

表 16-24 工件坐标系

类型	参数符号	参数名称	功　　能
动作	WKnCORD n：1～8	工件坐标系	设置工件坐标系
	WKnWO	工件坐标系原点	
	WKnWX	工件坐标系 X 轴位置点	
	WKnWY	工件坐标系 Y 轴位置点	
设置	可设置 8 个工件坐标系 参看图 16-10		

参数的编辑

参数名：WK4CORD　机器号：0

说明：Work coordinate 4

1：0.00　5：0.00
2：0.00　6：0.00
3：0.00
4：0.00

打印(P)　写入(W)　关闭(C)

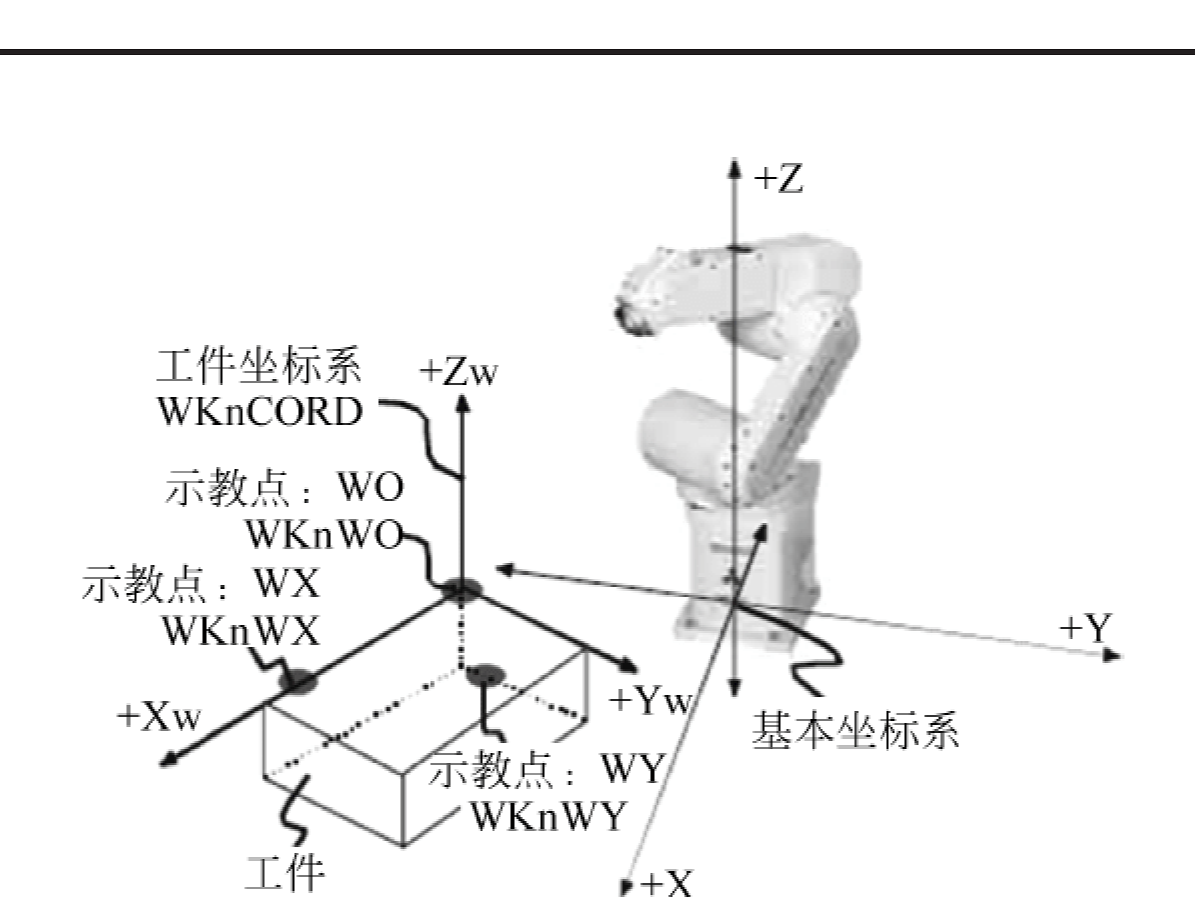


图 16-10 工件坐标系设置示意图

设置工件坐标系要注意以下两点。

(1) 工件坐标系的 X 轴、Y 轴方向最好要与“基本坐标系”一致。

(2) 工件坐标系原点只保证 X/Y/Z 轴坐标，不能满足 ABC 角度。

表 16-25 RETPATH

类型	参数符号	参数名称	功　　能
动作	RETPATH	程序中断执行 JOG 动作后的返回形式	设置程序中断执行 JOG 动作后的返回形式
设置	RETPATH＝0，无效；RETPATH＝1，以关节插补返回；RETPATH＝2，以直交插补返回		

在程序执行过程中，可能遇到不能满足工作要求的程序段，需要在线修改，系统提供了在中断后用 JOG 方式修改的功能。本参数设置在 JOG 修改完成后返回原自动程序的

形式。

图 16-11 是一般形式，图 16-12 是在"连续轨迹运行 CNT 模式"下的返回轨迹。

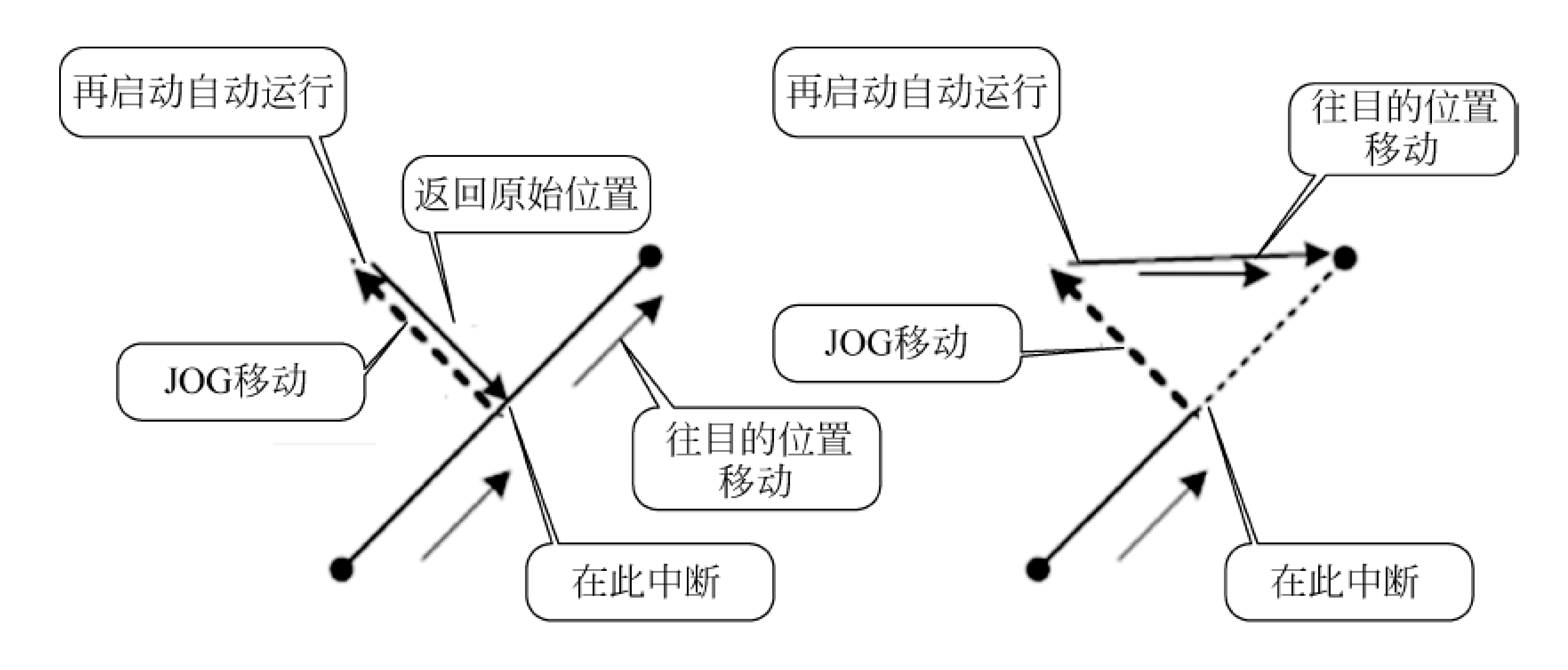


图 16-11 在自动程序中断进行 JOG 修正后返回的轨迹

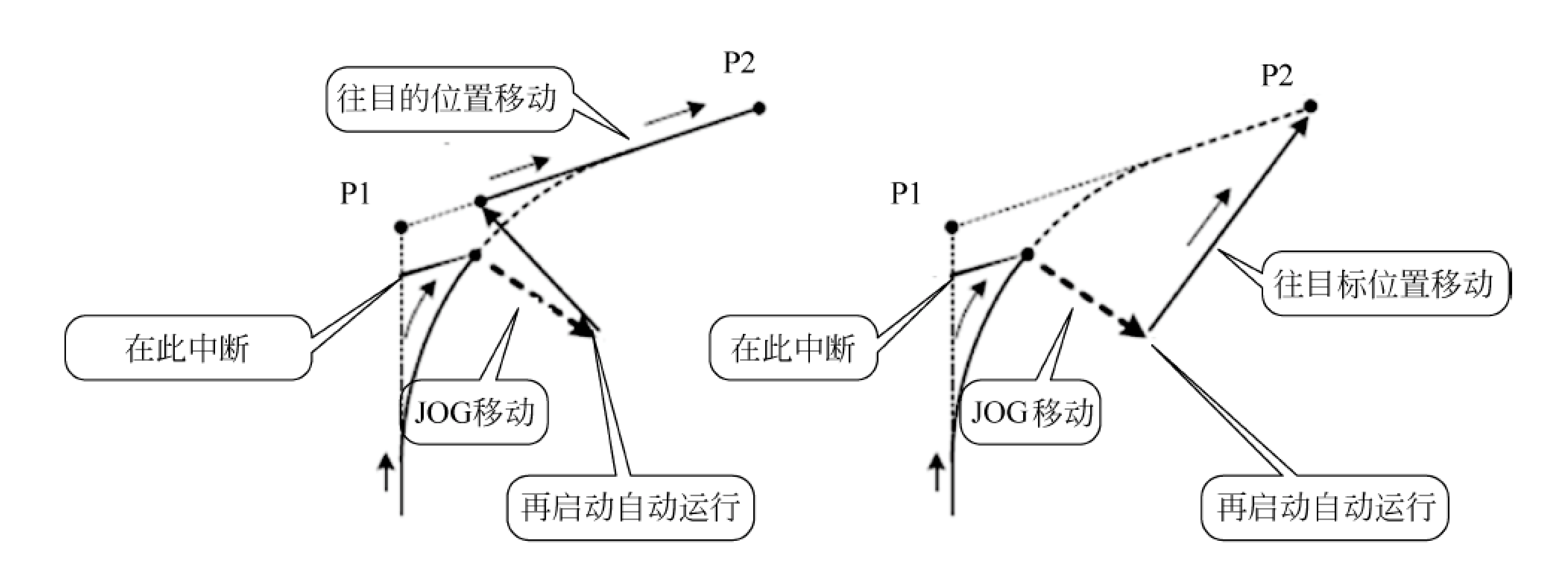


图 16-12 工件在"连续轨迹运行 CNT 模式"下的返回轨迹

表 16-26 重力方向

类型	参数符号	参数名称	功 能
动作	MEGDIR	重力在各轴方向上的投影值	设置重力在各轴方向上的投影值
设置	参见图 16-13		

由于安装方位的影响，重力加速度在各轴的投影值不同，如图 16-13 所示，所以要分别设置。

以图 16-13 倾斜 30°为例：

X 轴重力加速度(Xg) ＝ 9.8×sin(30°) ＝ 4.9。

Z 轴重力加速度(Zg) ＝ 9.8×cos(30°) ＝ 8.5。

因为 Z 轴与重力方向相反，所以为－8.5。

Y 轴重力加速度(Yg) ＝ 0.0。

所以设定值为(3.0，4.9，0.0，－8.5)。

安装姿势	设定值 (安装姿势，X轴的重力加速度， Y轴的重力加速度，Z轴的重力加速度)
放置地板 (标准)	(0.0, 0.0, 0.0, 0.0)
壁挂	(1.0, 0.0, 0.0, 0.0)
垂吊	(2.0, 0.0, 0.0, 0.0)
任意的姿势*1	(3.0, ***, ***, ***)

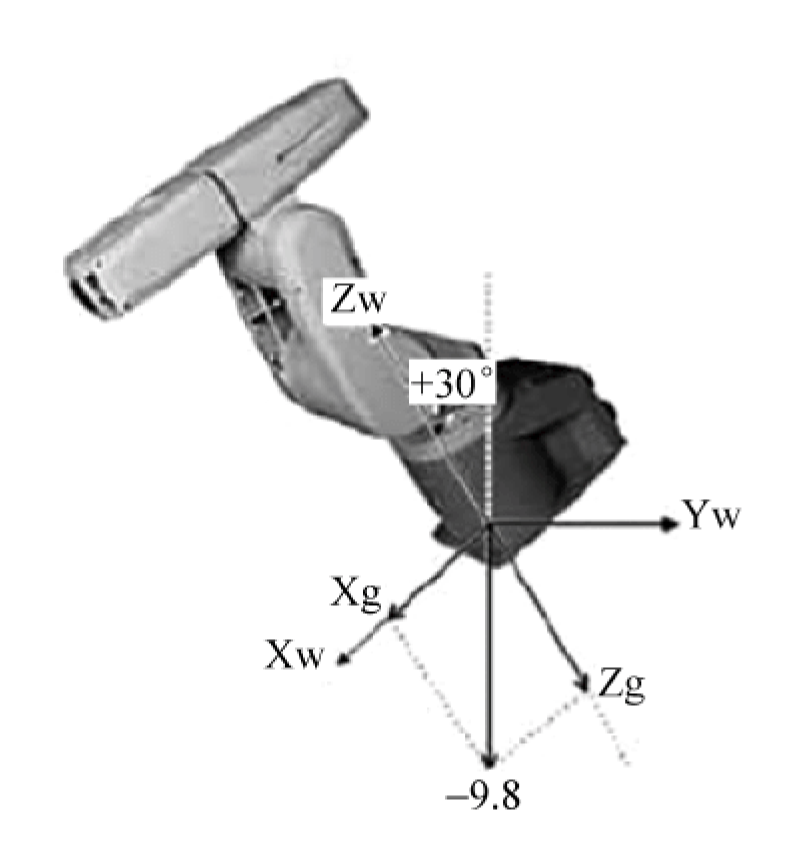


图 16-13　重力在各轴方向上的投影

表 16-27　ACCMODE

类型	参数符号	参数名称	功　　能
动作	ACCMODE	最佳加减速模式	设置上电后是否选择最佳加减速模式
设置	ACCMODE＝0,无效；ACCMODE＝1,有效		
如图 16-14 所示			

表 16-28　JADL

类型	参数符号	参数名称	功　　能
动作	JADL	最佳加减速倍率	设置最佳加减速倍率
设置			
如图 16-14 所示			

表 16-29　CMPERR

类型	参数符号	参数名称	功　　能
动作	CMPERR	伺服柔性控制报警选择	设置伺服柔性控制报警选择
设置	CMPERR＝0,不报警；CMPERR＝1,报警		
参见图 16-14			

图 16-14　最佳加减速模式及重力影响参数的设置

表 16-30　COL

类型	参数符号	参数名称	功　　能
动作	COL	碰撞检测	设置碰撞检测功能
	COLLVL	碰撞检测级别	1%～500%
	COLLVLJG	JOG 运行时的碰撞检测级别	1%～500%
设置	数值越小,灵敏度越高		
参见图 16-15			



图 16-15　碰撞检测相关参数的设置

需要做以下设置。

(1) 设置碰撞检测功能COL功能的有效/无效；

(2) 上电后的初始状态下碰撞检测功能COL功能的有效/无效；

(3) JOG操作中，碰撞检测功能COL功能的有效/无效。

COLLVL：自动运行时的碰撞检测的量级。

COLLVLJG：JOG运行时的碰撞检测的量级。

表16-31 预热运行

类型	参数符号	参数名称	功　　能
动作	WUPENA	预热运行模式	
	设置	WUPENA=0 无效； WUPENA=1 有效	
	WUPAXIS	预热运行对象轴	
	设置	bit ON 对象轴 bit OFF 非对象轴	
	WUPTIME	预热运行时间	
	设置	单位：分(1～60)	
	WUPOVRD	预热运行速度倍率	
设置	参见图16-16和图16-17		

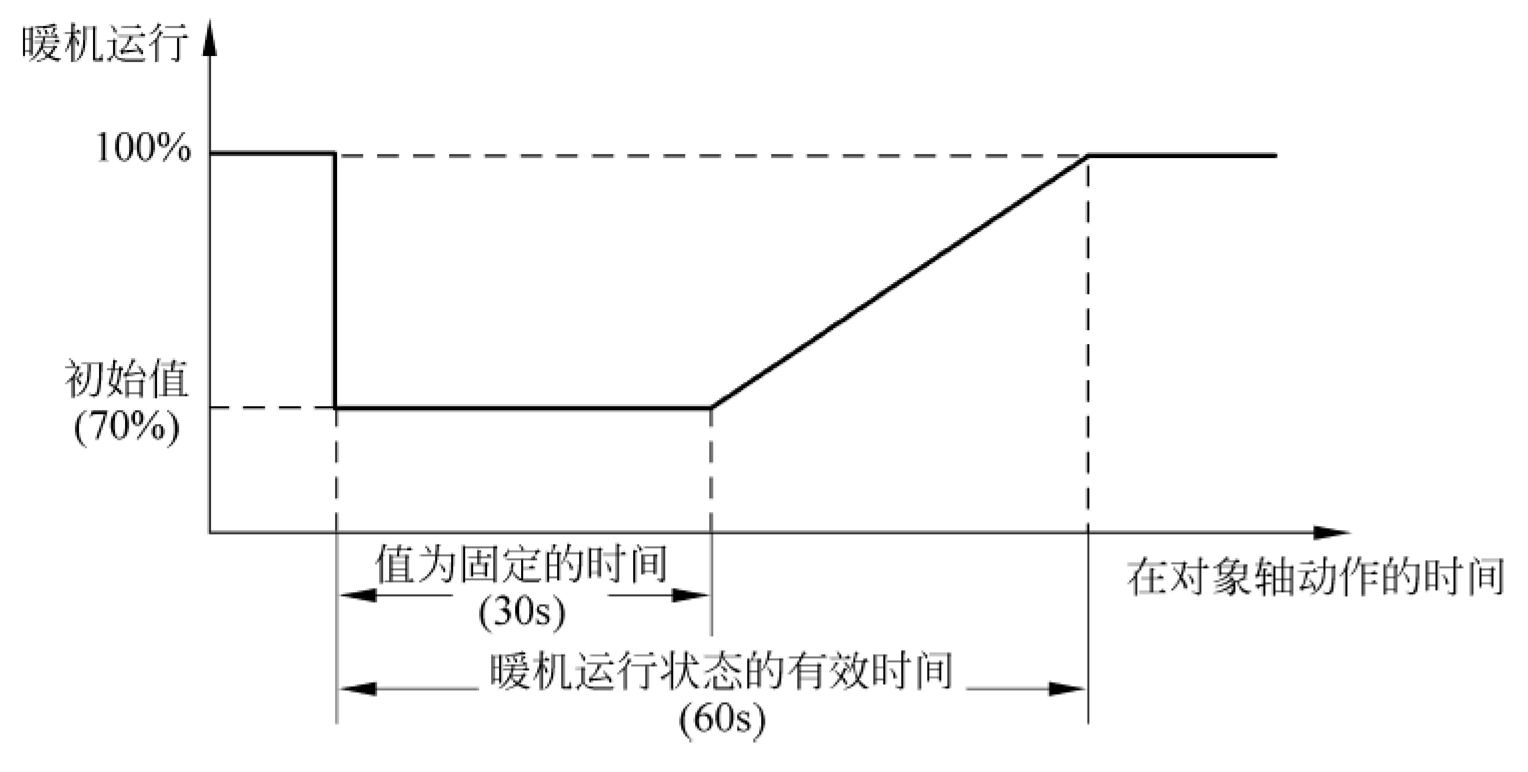


图16-16　预热运行的速度变化

在低温或长期停机后启动，需要进行预热运行(否则可能导致精度误差)。预热运行的本质是降低速度，实际通过降低速度倍率来实现。

设置参数如下。

WUPENA——设置预热模式有效/无效。0：无效；1：有效。

WUPAXIS——设置进入预热模式的轴。

WUPTIME——预热运行有效时间。

再启动时间——如果有某些轴预热后一直停止没有运行，经过设置时间后再次启动预热运行，这段时间就是“再启动时间”。

WUPOVRD——预热运行的速度倍率。

恒定值时间段比例——速度倍率为“恒定(直线段)的时间”相对“总预热有效时间”的



图 16-17　预热模式的相关参数设置

比例。

抓手参数的设置实际上是对使用电磁阀的设置。电磁阀分为单向电磁阀和双向电磁阀。可以直接使用外部输入输出信号控制电磁阀,在程序中直接发输入输出指令即可。

表 16-32　HIOTYPE

类型	参数符号	参数名称	功　　能
动作	HIOTYPE	抓手用电磁阀输入信号源型/漏型选择	
设置	HIOTYPE=0,源型;HIOTYPE=1,漏型		

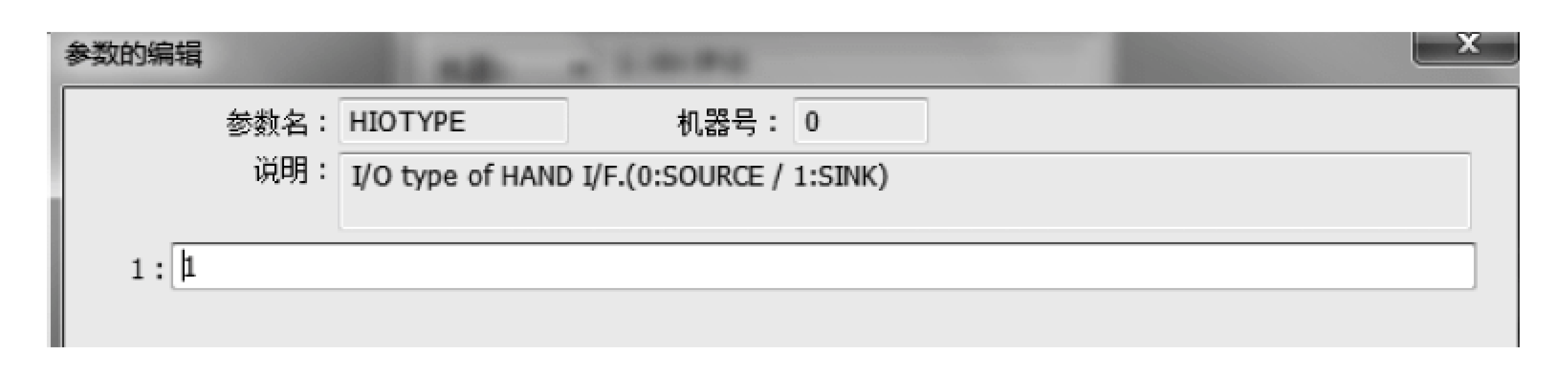


表 16-33　HANDTYPE

类型	参数符号	参数名称	功　　能
动作	HANDTYPE	设置电磁阀单线圈/双线圈及对应的外部信号	
设置	HANDTYPE=s**,单线圈;HANDTYPE=D**,双线圈;HANDTYPE=UMAC,特殊规格		

参数的编辑

参数名:HANDTYPE　机器号:1

说明:Control type for HAND1-8 (single/double/special=S***/D***/UMAC*)

1:	D900	5:	
2:	D902	6:	
3:	D904	7:	
4:	D906	8:	

参数 HANDTYPE 用于设置电磁阀的类型(单向、双向)和连接外部信号的地址号。

HANDTYPE＝D10——表示抓手1是双向电磁阀，外部输入信号地址为(10、11)。

HANDTYPE＝D10，D12——表示抓手1是双向电磁阀，外部输入信号地址为(10、11)；抓手2是双向电磁阀，外部输入信号地址为(12、13)。

HANDTYPE＝S10，S11，S12——表示抓手1是单向电磁阀，外部输入信号地址为(10)；抓手2是单向电磁阀，外部输入信号地址为(11)。

抓手3是单向电磁阀，外部输入信号地址为(13)。

表 16-34　HANDINIT

类型	参数符号	参数名称	功　　能
动作	HANDINIT	气动抓手的初始界面状态	
设置			

参数的编辑

参数名：HANDINIT　机器号：1

说明：Initial status for air hand I/F

1：1　5：1
2：0　6：0
3：1　7：1
4：0　8：0

HANDINIT——表示上电时，各抓手的“开”或“关”状态。

出厂设置如下：

抓手的种类	状态	输出信号号码的状态		
		机器 1	机器 2	机器 3
安装气动抓手 1/F 时(假设为双线螺管)	抓手 1＝开	900＝1	910＝1	920＝1
		901＝0	911＝0	921＝0
	抓手 2＝开	902＝1	912＝1	922＝1
		903＝0	913＝0	923＝0
	抓手 3＝开	904＝1	914＝1	924＝1
		905＝0	915＝0	925＝0
	抓手 4＝开	906＝1	916＝1	926＝1
		907＝0	917＝0	927＝0

如图 16-18 所示设置为上电后，各抓手全部为“开状态”。



图 16-18　参数 HANDINIT 的设置

如图 16-19 所示设置为上电后,1 号抓手为“关状态”。上电以后的初始状态关系到安全性,如果上电后抓手打开,可能会造成原来夹持的工件掉落,使用设置时必须特别注意。

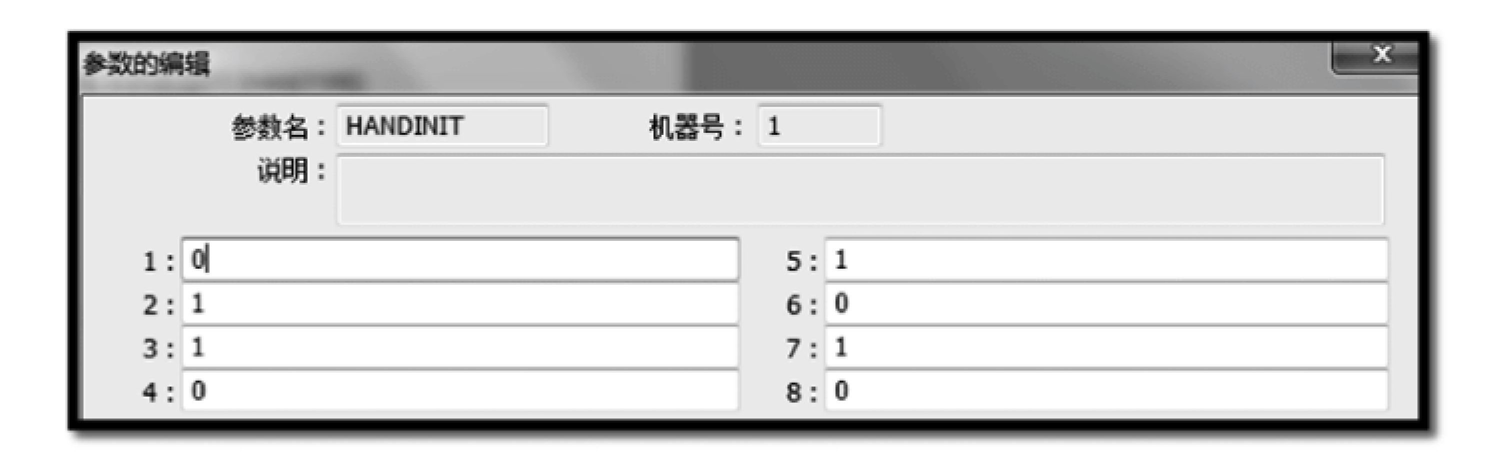


图 16-19　参数 HANDINIT 的设置

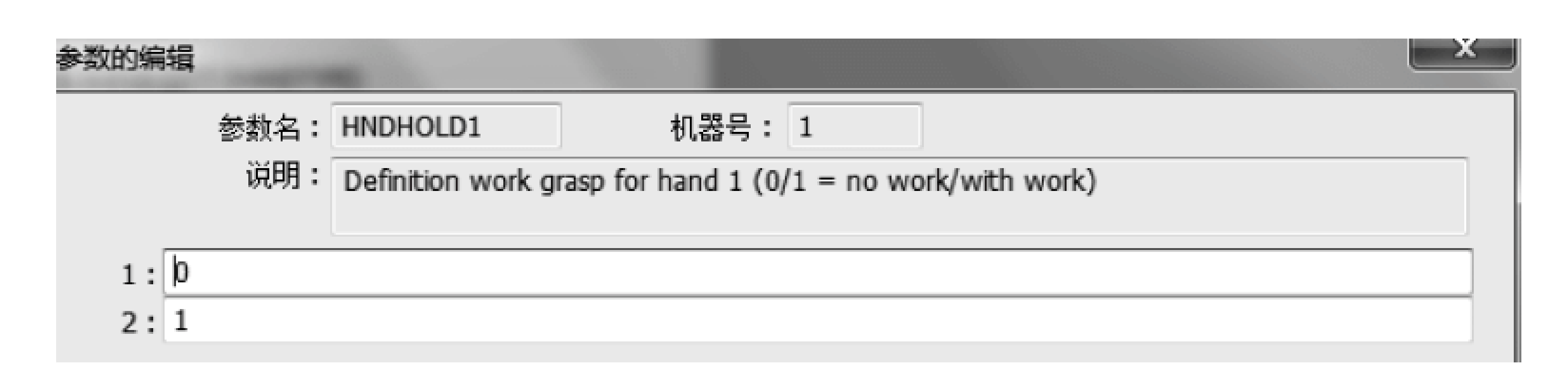
主要是信号地址的设置,在图 16-20 中控制电磁阀的信号——外部 I/O 卡上的输出信号地址是 12。在自动程序中使用 HOpen 1/HClose 1 指令就可以直接控制抓手动作。



图 16-20　抓手相关参数的设置

表 16-35　HNDHOLD1

类型	参数符号	参数名称	功　　能
动作	HNDHOLD1	抓手开状态与夹持工件关系	
设置	HNDHOLD1=0,不夹持工件;HNDHOLD1=1,夹持工件		



本参数是指在抓手“打开”状态下,是夹持工件还是不夹持工件(即工作方式是外张式还是抓紧式)。

16.2 程序型参数

16.2.1 程序型参数一览表

如表 16-36 所示是程序型参数一览表。

表 16-36 程序型参数一览表

序号	参数类型	参数符号	参数名称	参数功能
1	程序	SLT *	任务区内的程序名、运行模式、启动条件、执行程序行数	用于设置每一任务区内的程序名、运行模式、启动条件、执行程序行数
2	程序	TASKMAX	多任务个数	设置同时执行程序的个数
3	程序	SLOTON	程序选择记忆	设置已经选择的程序是否保持
4	程序	CTN	继续运行功能	
5	程序	PRGMDEG	程序内位置数据旋转部分的单位	
6	程序	PRGGBL	程序保存区域大小	
7	程序	PRGUSR	用户基本程序名称	
8	程序	ALWENA	特殊指令允许执行选择	选择一些特殊指令是否允许执行
9	程序	JRCEXE	JRC 指令执行选择	设置 JRC 指令是否可以执行
10	程序	JRCQTT	JRC 指令的单位	设置 JRC 指令的单位
11	程序	JRCORG	JRC 指令后的原点	设置 JRC 0 时的原点位置
12	程序	AXUNT	附加轴使用单位选择	设置附加轴的使用单位
13	程序	UER1～UER20	用户报警信息	编写用户报警信息
14	程序	RLNG	机器人使用的语言	设置机器人使用的语言
15	程序	LNG	显示用语言	设置显示用语言
16	程序	PST	程序号选择方式 是用外部信号选择程序的方法	在 START 信号输入的同时，使"外部信号选择的程序号"有效
17	程序	INB	STOP 信号改"B 触点"	可以对 stop、stopl、skip 信号进行修改
18	程序	ROBOTERR	EMGOUT 对应的报警类型和级别	设置 EMGOUT 报警接口对应的报警类型和级别

16.2.2 程序参数详解

程序参数是指与执行程序相关的参数，如表 16-37～表 16-55 所示。

任务区设置(插槽区 task slot)，如表 16-37 所示。

表 16-37 任务区设置

类型	参数符号	参数名称	功　　能
程序	SLT *	任务区内的程序名、运行模式、启动条件、执行程序行数	用于设置每一任务区内的程序名、运行模式、启动条件、执行程序行数
设置	参见图 16-21		

本参数用于设置每一任务区内的“程序名”、运行模式、启动条件、执行程序行数。

设置内容如下。

(1) 程序名：只能用大写字母，小写不识别。

(2) 运行模式：REP/CYC。

① REP——程序连续循环执行。

② CYC——程序单次执行。

(3) 启动条件：START/ALWAYS/ERROR。

① START：由 start 信号启动。

② ALWAYS：上电立即启动。

③ ERROR：发生报警时启动(多用于报警应急程序，不能执行有关运动的动作)。



图 16-21　任务区的设置

表 16-38　TASKMAX

类型	参数符号	参数名称	功　　能
程序	TASKMAX	多任务个数	设置同时执行程序的个数
设置	初始值：8		



同时执行程序，只可能一个是动作程序，其余为数据信息处理程序，这样就不会出现混乱动作的情况。

表 16-39　程序选择记忆

类型	参数符号	参数名称	功　　能
动作	SLOTON	程序选择记忆	设置已经选择的程序是否保持
设置	SLOTON=0　记忆无效　非保持 SLOTON=1　记忆有效　非保持 SLOTON=2　记忆无效　保持 SLOTON=3　记忆有效　保持		
参见图 16-22			

本参数用于设置选择程序在断电-上电后是否保持原来的选择状态。设置方式参见图 16-22。

记忆：断电-上电后保持原来选择程序(在任务区 1 内)。

保持：程序循环执行结束后是否保持原程序名。0：不保持；1：保持。



图 16-22　参数 SLOTON 设置

表 16-40　CTN

类型	参数符号	参数名称	功　　能
程序	CTN	继续工作功能	在程序执行过程中，如果断电，则保存所有工作状态，在上电后从断电处开始执行(因此必须特别注意安全)。视觉指令不支持这一功能
设置	CTN＝0，无效；CTN＝1，有效		

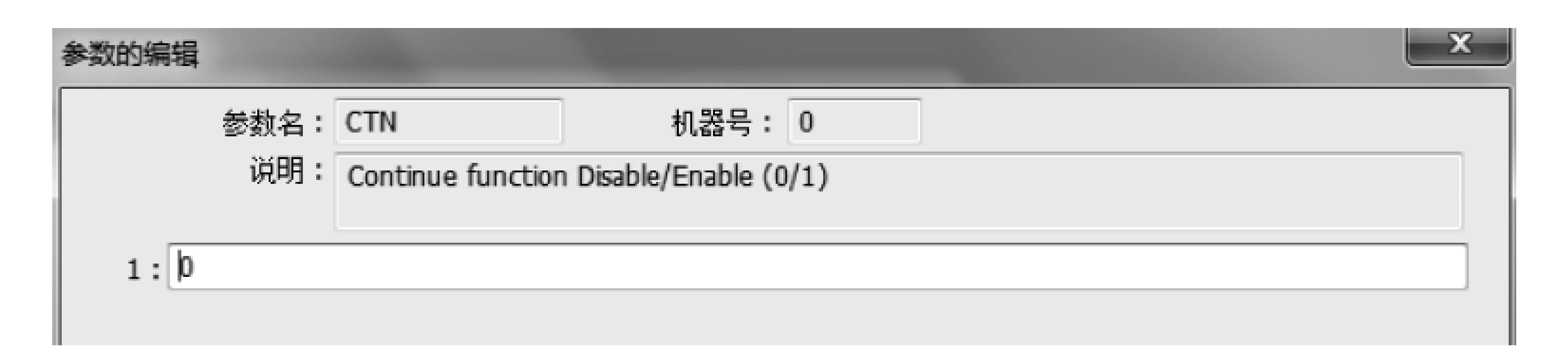


表 16-41　PRGMDEG

类型	参数符号	参数名称	功　　能
动作	PRGMDEG	程序内位置数据旋转部分的单位	
设置	PRGMDEG＝0 时单位为 RAD(弧度)，PRGMDEG＝1 时单位为 DEG(度)		

每一点的位置数据为(X，Y，Z，A，B，C)，其中，A/B/C 为旋转轴部分。本参数用于设置 A/B/C 旋转轴的单位是“弧度”还是“度”。初始设置为度(deg)。

表 16-42　PRGGBL

类型	参数符号	参数名称	功　　能
动作	PRGGBL	程序保存区域大小	本参数用于设置程序保存区域的大小
设置	PRGGBL＝0，标准型；PRGGBL＝1，扩展型		



表 16-43　用户基本程序名称

类型	参数符号	参数名称	功　　能
动作	PRGUSR	用户基本程序名称	设置用户基本程序名称
设置	字符		
参数的编辑 参数名：PRGUSR　机器号：0 说明：User base program name 1：			
用户基本程序是定义“全局变量”的程序，内容仅为 Def Inte 或 Dim			

表 16-44　ALWENA

类型	参数符号	参数名称	功　　能
动作	ALWENA	特殊指令允许执行选择	规定一些特殊指令是否允许执行
设置	ALWENA＝0，不可执行；ALWENA＝1，可执行。 对于上电就启动执行的程序简称为“上电执行程序”，在“上电执行程序”中，某些特殊指令 Xrun、Xload、Xstp、Servo、Xrst、Reset Error 是否能够执行需要通过本参数设置		
参数的编辑 参数名：ALWENA　机器号：1 说明： 1：0			

表 16-45　JRCEXE

类型	参数符号	参数名称	功　　能
动作	JRCEXE	JRC 指令执行选择	设置 JRC 指令是否可以执行 JRCEXE＝0，不可执行； JRCEXE＝1，可执行
	JRCQTT	JRC 指令的单位	设置 JRC 指令的单位
	JRCORG	JRC 指令后的原点	JRCORG 设置 JRC 0 时的原点位置
设置			
参数的编辑 参数名：JRCEXE　机器号：1 说明： 1：0			

表 16-46　AXUNT

类型	参数符号	参数名称	功　　能
动作	AXUNT	选择附加轴使用单位	设置附加轴的使用单位
设置	AXUNT＝0，角度(deg)；AXUNT＝1，长度(mm)		

续表



表 16-47　UER

类型	参数符号	参数名称	功　　能
程序	UER1～UER20	用户报警信息	编写用户报警信息
设置	用户自行编制的"报警信息"		



表 16-48　RLNG

类型	参数符号	参数名称	功　　能
程序	RLNG	机器人使用的语言	设置机器人使用的语言
设置	RLNG＝2，MELFA-BASIC V； RLNG＝1，MELFA-BASIC IV		



表 16-49　LNG

类型	参数符号	参数名称	功　能
程序	LNG	显示用语言	设置显示用语言
设置	LNG＝JPN,日语；LNG＝ENG,英语		



表 16-50　PST

类型	参数符号	参数名称	功　能
程序	PST	程序号选择方式 用外部信号选择程序的方法	在 START 信号输入的同时,使外部信号选择的程序号有效
设置	PST＝0,无效；PST＝1,有效		

参数的编辑
参数名： PST
机器号： 0
说明： Prog. No. read starting (1:Enable,0:Disable)
1： 0

表 16-51　INB

类型	参数符号	参数名称	功　能
信号	INB	STOP 信号改“B 触点”	可以对 stop、stopl、skip 信号进行修改
设置	INB＝0,A 触点；INB＝1,B 触点		

参数的编辑
参数名： INB
机器号： 0
说明： Stop input signal normaly open(0)/close(1)
1： 0

表 16-52　ROBOTERR

类型	参数符号	参数名称	功　能
信号	ROBOTERR	EMGOUT 对应的报警类型和级别	设置 EMGOUT 报警接口对应的报警类型和级别
设置	通常设置为“7”		

参数名： ROBOTERR
机器号： 0
说明： Bit pattern of robot error output signal setting (0-7:C/L/H)
1： 7

表 16-53　E7730

类型	参数符号	参数名称	功　　能
动作	E7730	解除 CCLINK 报警	
设置	E7730＝0，不可解除；E7730＝1，可解除		

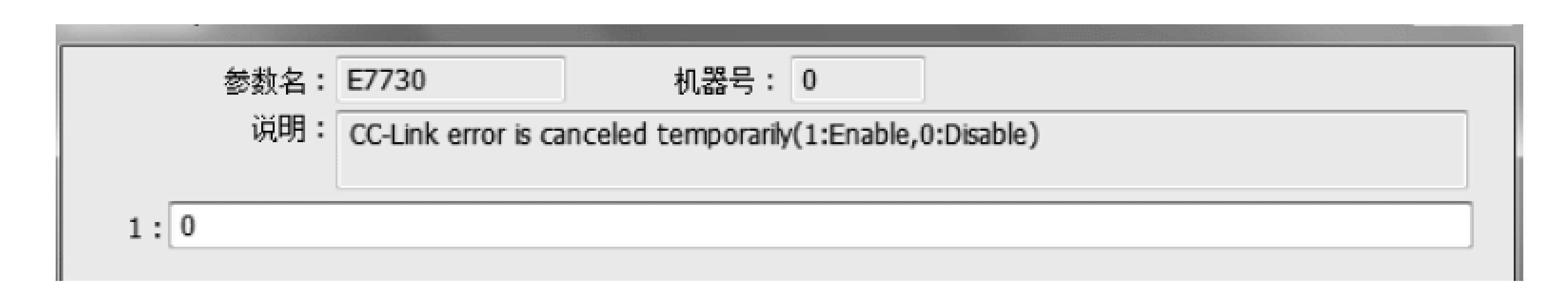


表 16-54　ORST0

类型	参数符号	参数名称	功　　能
动作	ORST0	输出信号的复位模式	设置当 CLR 指令或 OUTRESET 信号时，输出信号如何动作
设置	参见图 16-23 和图 16-24		

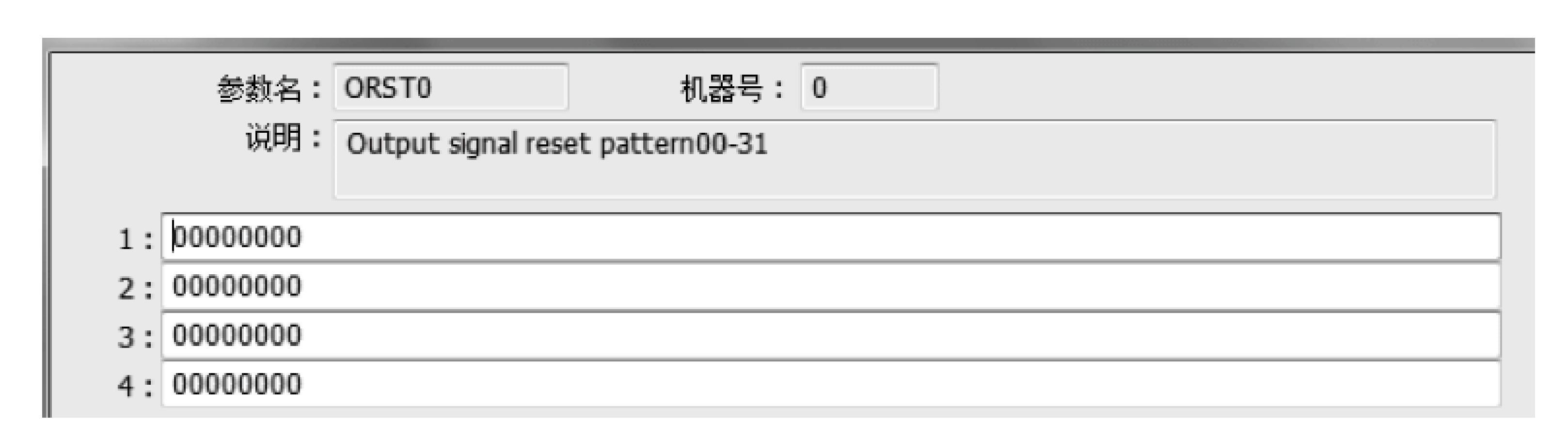




图 16-23　输出信号复位模式参数



图 16-24　ORST0 参数的设置

“保持”的含义是保持输出信号原来的状态。即复位前为 ON，就为 ON，复位前为 OFF，就为 OFF。

表 16-55　SLRSTIO

类型	参数符号	参数名称	功　　能
动作	SLRSTIO	程序复位时是否执行输出信号的复位	
设置	SLRSTIO＝0，不执行；SLRSTIO＝1，执行		

续表

参数名：SLRSTIO　机器号：0 说明：Output signal reset with SLOTINIT (1:ON, 0:OFF) 1：0

16.3　需要思考的问题

（1）动作型参数主要包含哪些方面的内容？

（2）程序型参数主要包含哪些方面的内容？

（3）如何设置机器人在直角坐标系内的行程范围？

（4）什么是基本坐标系偏置？如何设置基本坐标系偏置？

（5）使用什么参数设置任务区内的“程序名”“运行模式”“启动条件”“执行程序行数”？

（6）如何设置程序号选择方式？

第 17 章　第 17 日——参数的功能及设置(2)

【学习目的】

本章要继续学习机器人所使用的操作型参数和网络通信参数。

17.1　操作型参数

17.1.1　操作参数一览表

如表 17-1 所示是操作型参数一览表。

表 17-1　操作型参数一览表

序号	参数符号	参数名称及功能	出厂值
1	BZR	设置报警时蜂鸣器音响 OFF/ON	1(ON)
2	PRSTENA	程序复位操作权 设置“程序复位操作”是否需要操作权	0(必要)
3	MDRST	随模式转换进行程序复位	0(无效)
4	OPDISP	操作面板显示模式	
5	OPPSL	操作面板为 AUTO 模式时的程序选择操作权	1(OP)
6	RMTPSL	操作面板的按键为 AUTO 模式时的程序选择操作权	0 (外部)
7	OVRDTB	示教单元上改变速度倍率的操作权选择(不必要=0,必要=1)	1(必要)
8	OVRDMD	模式变更时的速度设定	
9	OVRDENA	改变速度倍率的操作权(必要=0,不必要=1)	0(必要)
10	ROMDRV	切换程序的存取区域	
11	BACKUP	将 RAM 区域的程序复制到 ROM 区	
12	RESTORE	将 ROM 区域的程序复制到 RAM 区	
13	MFINTVL	维修预报数据的时间间隔	
14	MFREPO	维修预报数据的通知方法	
15	MFGRST	维修预报数据的复位	
16	MFBRST	维修预报数据的复位	
17	DJNT	位置回归相关数据	
18	MEXDTL	位置回归相关数据	
19	MEXDTL1～5	位置回归相关数据	
20	MEXDBS	位置回归相关数据	
21	TBOP	是否可以通过示教单元进行程序启动	

17.1.2 操作参数详解

操作参数详解见表 17-2～表 17-18。

表 17-2 BZR

类型	参数符号	参数名称	功　　能
操作	BZR	报警时蜂鸣器音响 OFF/ON	设置报警时蜂鸣器音响
设置	OFF＝0ON＝1		



表 17-3 PRSTENA

类型	参数符号	参数名称	功　　能
操作	PRSTENA	程序复位操作权	设置“程序复位操作”是否需要操作权
设置	必要＝0，不要＝1，出厂值 0（必要）。 如果设置为不要操作权，就可在任何位置使程序复位，有安全上的危险。特别是不能在示教单元上使程序复位		

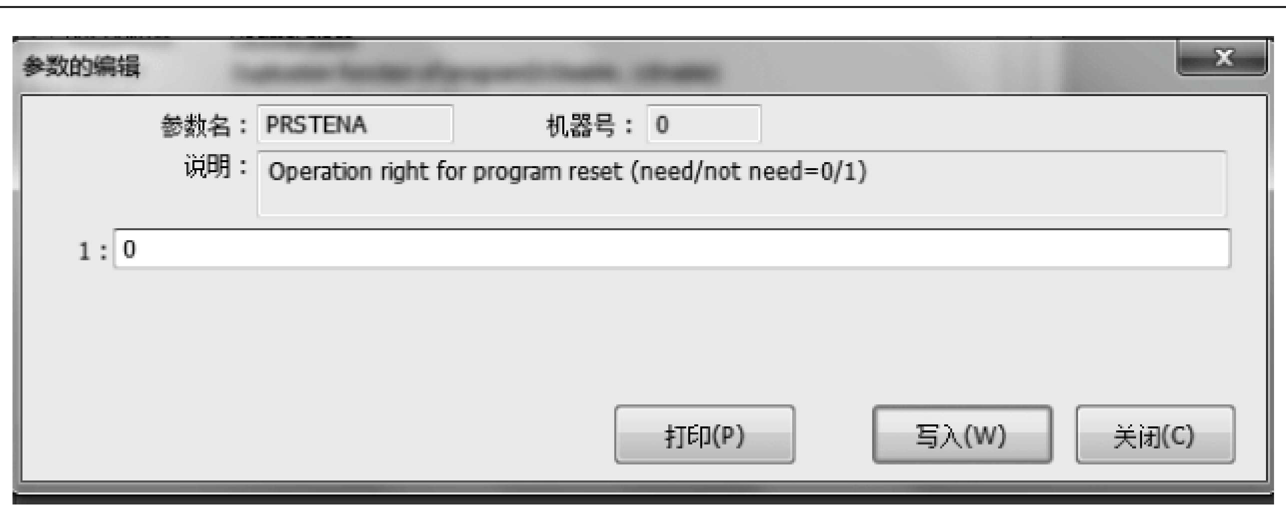


表 17-4 MDRST

类型	参数符号	参数名称	功　　能
操作	MDRST	随模式转换进行程序复位	随模式转换进行程序复位
设置	无效＝0，有效＝1，出厂值 0（无效）		

表 17-5 OPDISP

类型	参数符号	参数名称	功　　能
操作	OPDISP	操作面板显示模式	设置模式切换时的显示内容
设置	OPDISP＝0,显示速度倍率；OPDISP＝1,显示原内容		



表 17-6 OPPSL

类型	参数符号	参数名称	功　　能
操作	OPPSL	操作面板上已经选择 AUTO 模式时的程序选择操作权	操作面板为 AUTO 模式时的程序选择操作权
设置	OPPSL＝0,外部信号(指来自外部 I/O 的信号)；OPPSL＝1,OP 操作面板		



表 17-7 RMTPSL

类型	参数符号	参数名称	功　　能
操作	RMTPSL	AUTO 模式时的程序选择操作权	由外部信号选择 AUTO 模式时的程序选择操作权
设置	外部＝0,OP＝1,出厂值 0(外部)		



表 17-8　OVRDTB

类型	参数符号	参数名称	功　　能
操作	OVRDTB	示教单元上改变速度倍率的操作权选择	设置示教单元上改变速度倍率的操作权选择
设置	不必要＝0,必要＝1,出厂值 1		



表 17-9　OVRDMD

类型	参数符号	参数名称	功　　能
操作	OVRDMD	模式变更时的速度设定	在示教模式变更为自动模式、自动模式变更为示教模式时自动设置的速度倍率
设置	第 1 栏：在示教模式变更为自动模式时自动设置的速度倍率； 第 2 栏：在自动模式变更为示教模式时自动设置的速度倍率。 设置数据＝0,保持原来的速度倍率		



表 17-10　OVRDENA

类型	参数符号	参数名称	功　　能
操作	OVRDENA	改变速度倍率的操作权	设置改变速度倍率是否需要操作权
设置	必要＝0,不必要＝1；出厂值 0(必要)		



表 17-11　ROMDRV

类型	参数符号	参数名称	功　　能
操作	ROMDRV	切换程序的存取区域	将程序的存取区域在 RAM/ROM 之间切换
设置	0＝RAM 模式(初始值使用 SRAM) 1＝ROM 模式 2＝高速 RAM 模式(使用 DRAM) 出厂值＝2		

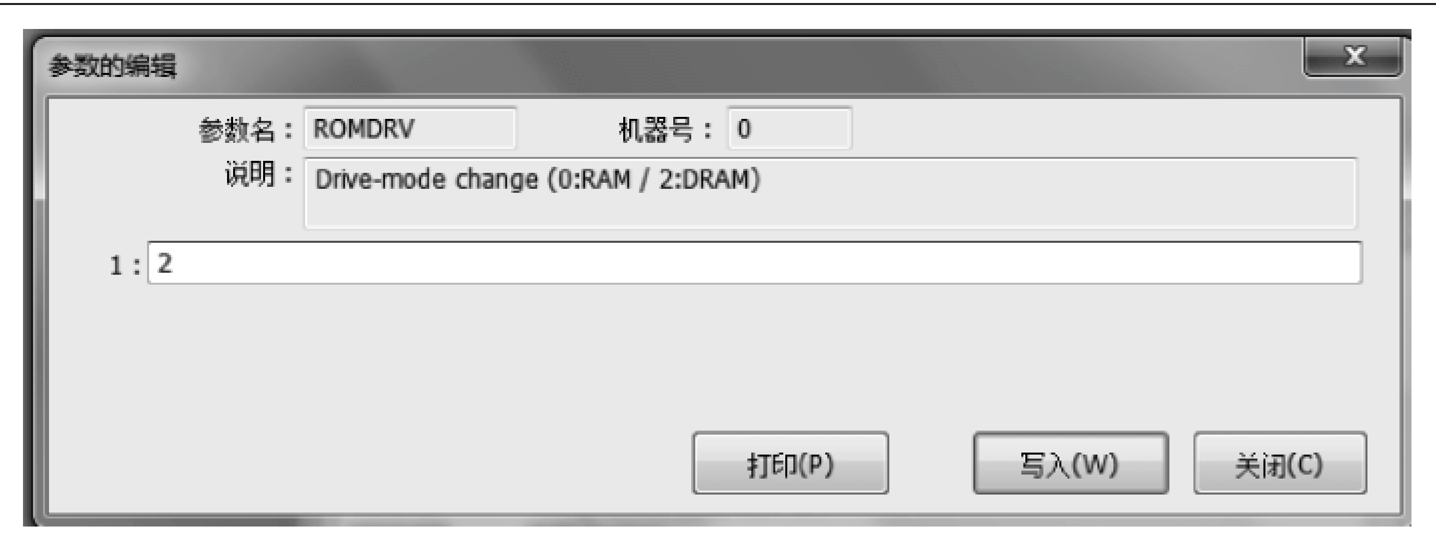


表 17-12　BACKUP

类型	参数符号	参数名称	功　　能
操作	BACKUP	将 RAM 区域的程序复制到 ROM 区	将程序、参数、共变量从 RAM 区域复制到 ROM 区
设置	如下图		



表 17-13　RESTORE

类型	参数符号	参数名称	功　　能
操作	RESTORE	将 ROM 区域的程序复制到 RAM 区	将程序、参数、共变量从 ROM 区域复制到 RAM 区
设置	FLROM→SRAM		



表 17-14　MFINTVL

类型	参数符号	参数名称	功　　能
操作	MFINTVL	维修预报数据的时间间隔	设置维修预报数据的时间间隔
设置	第 1 栏：采样量级(1～5 小时)； 第 2 栏：维修预报数据的时间间隔(1～24 小时)		

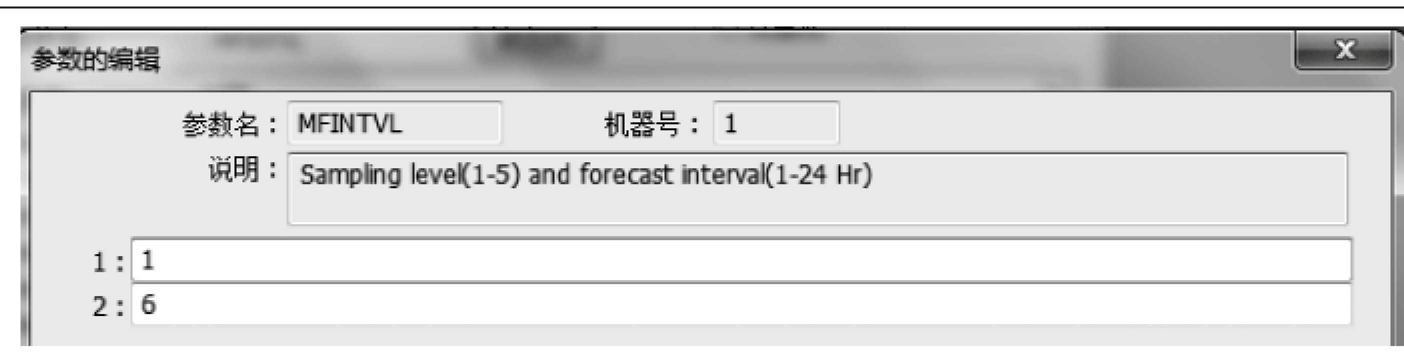


表 17-15　MFREPO

类型	参数符号	参数名称	功　　能
操作	MFREPO	维修预报数据的通知方法	
设置	第 1 栏：发出报警＝1,不发出报警＝0 第 2 栏：专用信号输出＝1,专用信号不输出＝0		

参数的编辑
参数名：MFREPO　机器号：1
说明：Warning generation and signal output for M.F.(effective=1/invalidity=0)
1：1
2：0

表 17-16　MFGRST

类型	参数符号	参数名称	功　　能
操作	MFGRST	维修预报数据的复位	将润滑油数据复位
设置	0＝全部轴复位 1～8＝指定轴复位		



表 17-17　MFBRST

类型	参数符号	参数名称	功　　能
操作	MFBRST	维修预报数据的复位	将皮带数据复位
设置	0＝全部轴复位 1～8＝指定轴复位		

参数的编辑
参数名：MFBRST　机器号：1
说明：
1：0

表 17-18　TBOP

类型	参数符号	参数名称	功　　能
操作	TBOP	是否可以通过示教单元进行程序启动	设置是否可以通过示教单元进行程序启动
设置	0=不可以,1=可以		

参数的编辑
参数名： TBOP　机器号： 0
说明： Program start operation by TB. (0:Disable, 1:Enable)
1： 0

17.2　网络通信参数

17.2.1　通信及现场网络参数一览表

如表 17-19 所示是通信及现场网络参数一览表。

表 17-19　通信及现场网络参数一览表

序号	参数符号	参 数 名 称	参 数 功 能
1	COMSPEC	RT ToolBox2 通信方式	选择控制器与 RT ToolBox2 软件的通信模式
2	COMDEV	通信端口分配设置	
3	NETIP	控制器的 IP 地址	192.168.0.20
4	NETMSK	子网掩码	255.255.255.0
5	NETPORT	端口号码	—

17.2.2　通信及网络参数详解

通信及网络参数详解见表 17-20～表 17-21。

表 17-20　RS-232 通信参数

类型	参数符号	参数名称	功　　能
通信		RS-232 通信参数	
设置	如下图所示。		



表 17-21　以太网参数

类型	参数符号	参数名称	功　　能
通信		以太网参数	
设置	如下图所示。		



17.3　需要思考的问题

（1）操作型参数主要包含哪些方面的内容？

（2）通信型参数主要包含哪些方面的内容？

（3）如何设置程序复位操作权？

（4）如何设置 RS-232 通信参数？

（5）如何设置以太网通信参数？

第 18 章 第 18 日——参数的功能及设置(3)

【学习目的】

本章学习通过参数来设置“输入输出端子”的功能。

18.1 输入输出信号参数

18.1.1 专用输入输出信号参数一览表

如表 18-1 所示是专用输入输出信号一览表。

表 18-1 专用输入输出信号一览表

序号	参数类型	参数符号	参数名称	参数功能
1		AUTOENA	可自动运行	“自动使能”信号
2		Start	启动	程序启动信号。在多任务时,启动全部任务区内的程序
3		Stop	停止	停止程序执行。在多任务时,停止全部任务区内的程序。STOP 信号地址是固定的
4		STOP2	停止	功能与 STOP 信号相同。但输入信号地址可改变
5	输入	Slotinit	程序复位	解除程序中断状态,返回程序起始行。对于多任务区,指令所有任务区内的程序复位。但对以 ALWAYS 或 ERROR 为启动条件的程序除外
6		Errrset	报错复位	解除报警状态
7		Cycle	单(循环)运行	选择停止“程序连续循环”运行
8		Srvoff	伺服 OFF	指令全部机器人伺服电源=OFF
9		Srvon	伺服 ON	指令全部机器人伺服电源=ON
10		IOENA	操作权	外部信号操作有效
11	SAFEPOS	回退避点	“回退避点”启动信号。退避点由参数设置	
12	OUTRESET	输出信号复位	“输出信号复位”指令信号。复位方式由参数设置	

续表

序号	参数类型	参数符号	参数名称	参 数 功 能
13	MELOCK	机械锁定	程序运动，机器人机械不动作	
14	信号	PRGSEL	选择程序号	用于确认已经选择的程序号
15		OVRDSEL	选择速度倍率	用于确认已经选择的程序倍率
16		PRGOUT	请求输出程序号	请求输出程序号
17		LINEOUT	请求输出程序行号	请求输出程序行号
18		ERROUT	请求输出报警号	请求输出报警号
19		TMPOUT	请求输出控制柜内温度	请求输出控制柜内温度
20		IODATA	数据输入信号端地址	用一组输入信号端子表示选择的程序号或速度倍率(8421 码) 表示输出状态也是同样方法
21		JOGENA	选择 JOG 运行模式	JOGENA＝0，无效 JOGENA＝ 1，有效
22		JOGM	选择 JOG 运行的坐标系	JOGM ＝ 0/1/2/3/4 关节/直交/圆筒/三轴直交/工具
23		JOG＋	JOG＋指令信号	设置指令信号的起始/结束地址信号(8 轴)
24		JOG－	JOG－指令信号	设置指令信号的起始/结束地址信号(8 轴)
25		JOGNER	JOG 运行时不报警	在 JOG 运行时即使有故障也不发报警信号
26	SnSTART	各任务区程序启动信号(共 32 区)	设置各任务区程序启动信号地址	
27		SnSTOP	各任务区程序停止信号(共 32 区)	设置各任务区程序停止信号地址
28		SnSRVON	各机器人伺服 ON	设置各机器人伺服 ON
29		SnSRVOFF	各机器人伺服 OFF	设置各机器人伺服 OFF
30		SnMELOCK	(各机器人)机械锁定	设置(各机器人)机械锁定信号
31		MnWUPENA	各机器人预热运行模式选择	设置各机器人预热运行模式

18.1.2 专用输入输出信号详解

本节叙述机器人系统所具备的(内置)“输入”“输出”功能，通过参数可将这些功能设置到“输入输出端子”。在没有进行参数设置前，I/O 卡上的“输入输出端子”是没有功能定义的，就像一台空白的 PLC 控制器一样。

1. 通用输入输出 1

为了便于阅读和使用，将输入输出信号单独列出。在机器人系统中，专用输入输出的(功能)“名称(英文)”是一样的，即同一“名称(英文)”可能表示输入也可能表示输出，初学者刚开始阅读指令手册时会感到困惑，本书将输入输出信号单独列出，便于读者阅读和使用。

表 18-2 是输入信号功能一览表 1，这一部分信号是经常使用的。

表 18-2　输入信号功能一览表 1

类型	参数符号	参数名称	功　　能
输入	AUTOENA	可自动运行	“自动使能”信号
	Start	启动	程序启动信号。在多任务时，启动全部任务区内的程序
	Stop	停止	停止程序执行。在多任务时，停止全部任务区内的程序。Stop 信号地址是固定的
	Stop2	停止	功能与 Stop 信号相同，但输入信号地址可改变
	Slotinit	程序复位	解除程序中断状态，返回程序起始行。对于多任务区，指令所有任务区内的程序复位。但对以 ALWAYS 或 ERROR 为启动条件的程序除外
	Errrset	报错复位	解除报警状态
	Cycle	单(循环)运行	选择停止“程序连续循环”运行
	Srvoff	伺服 OFF	指令全部机器人伺服电源＝OFF
	Srvon	伺服 ON	指令全部机器人伺服电源＝ON
	IOENA	操作权	外部信号操作有效
设置	参见图 18-1		



图 18-1　通用输入输出 1 相关参数的设置

(1) AUTOENA——“自动使能”信号。AUTOENA＝1 允许选择自动模式。AUTOENA＝0 不允许选择自动模式，选择自动模式则报警(L5010)。但是如果不分配输入端子信号则不报警，所以一般不设置 AUTOENA 信号。

(2) Cycle——Cycle＝ON，程序只执行一次(执行到 END 即停止)。

(3) 伺服 ON 信号在“自动模式下”才有效，选择手动模式时无效。

(4) Stop——是一种暂停。Stop＝ON，程序停止。重新发 Start 信号，程序从断点启动。Stop 信号固定分配到“输入信号端子 0”。除了 Stop 信号，其他输入信号地址可以任意设置修改。例如，Start 信号可以从出厂值“3”改为“31”。

2. 通用输入输出 2

表 18-3 是输入信号功能一览表 2。由于在 RT 软件设置界面上是同一界面，所以将这些信号归为一类。

表 18-3　输入信号功能一览表 2

类型	参数符号	参数名称	功　　能
动作	SAFEPOS	回退避点	“回退避点”启动信号 退避点由参数设置
	OUTRESET	输出信号复位	“输出信号复位”指令信号。复位方式由参数设置
MELOCK	机械锁定	程序运动,机器人机械不动	
设置	参见图 18-2		



图 18-2　通用输入输出 2 相关参数的设置

3. 数据参数

表 18-4 是输入信号功能一览表 3。由于在 RT 软件设置界面上是同一界面,所以将这些信号归为一类。

表 18-4　输入信号功能一览表 3

类型	参数符号	参数名称	功　　能
信号	PRGSEL	选择程序号	用于确认输入的数据为程序号
	OVRDSEL	选择速度倍率	用于确认输入的数据为程序倍率
	PRGOUT	请求输出程序号	请求输出程序号
	LINEOUT	请求输出程序行号	请求输出程序行号
	ERROUT	请求输出报警号	请求输出报警号
	TMPOUT	请求输出控制柜内温度	请求输出控制柜内温度
	IODATA	数据输入信号端地址	用一组输入信号端子(8421 码)作为输入数据用 表示输出数据也是同样方法
设置	参见图 18-3		

PRGSEL——程序选择确认信号。当通过 IODATA 指定的输入端子(构成 8421 码)选择程序号后,将 PRGSEL=ON,即确认了输入的数据为程序号。

4. JOG 运行信号

这是不用示教单元而用外部信号实现 JOG 运行的输入输出端子设置参数。

表 18-5 是 JOG 运行输入信号功能一览表。由于在 RT 软件设置界面上是同一界面,所以将这些信号归为一类。

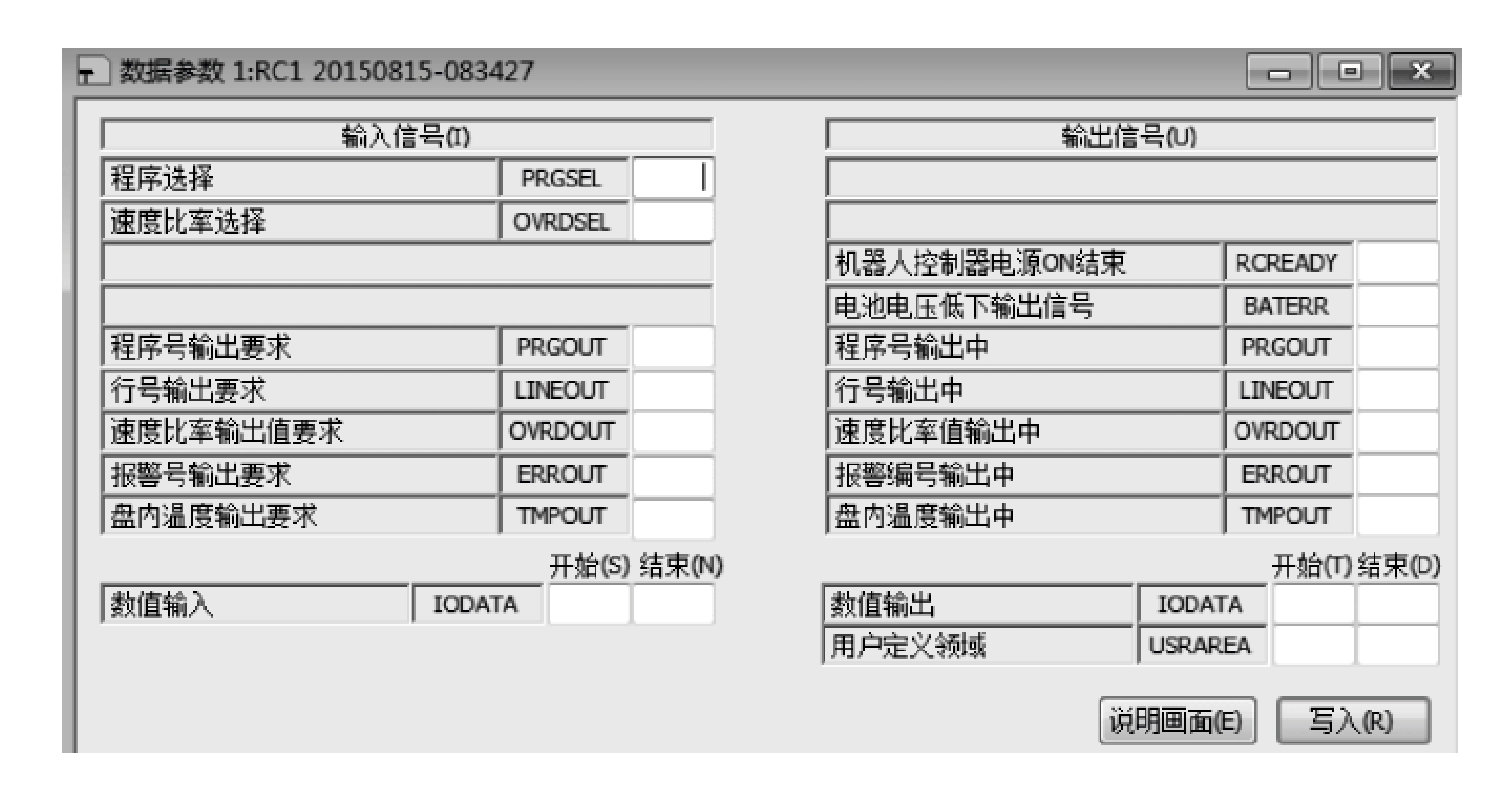


图 18-3　数据参数的设置

表 18-5　JOG 运行输入信号功能一览表

类型	参数符号	参数名称	功　　能
信号	JOGENA	选择 JOG 运行模式	JOGENA＝0，无效；JOGENA＝1，有效
	JOGM	选择 JOG 运行的坐标系	JOGM＝0/1/2/3/4，关节/直交/圆筒/三轴直交/工具
	JOG＋	JOG＋指令信号	设置指令信号的起始/结束地址信号（8 轴）
	JOG－	JOG－指令信号	设置指令信号的起始/结束地址信号（8 轴）
	JOGNER	JOG 运行时不报警	在 JOG 运行时即使有报警也不发报警信号
设置			

执行外部信号做 JOG 运动的方法如下。

(1) 选择“自动模式”(只有在自动模式下，伺服 ON 才有效)。

(2) 发“伺服 ON”＝1 信号。

(3) 使 JOGENA＝ 1(在图 18-4 中为输入端子 16)(在图 18-4 中输入端子 24～29 为 J1～J6 的 JOG＋信号)，发出各轴 JOG＋信号，各轴做 JOG 动作。

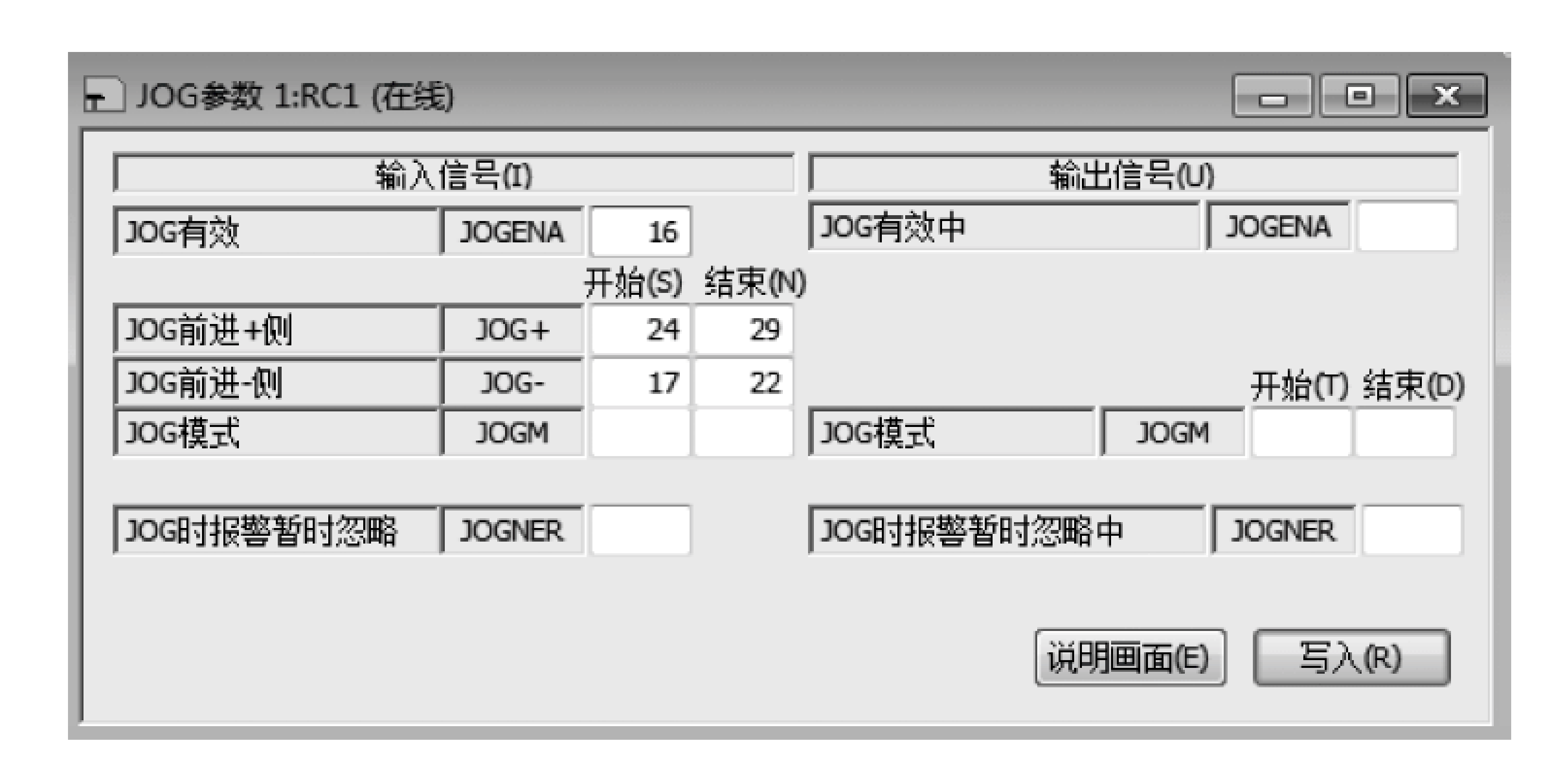


图 18-4　JOG 运行相关参数的设置

5. 各任务区启动信号

各任务区启动信号如表 18-6 所示。

表 18-6　各任务区启动信号

类型	参数符号	参数名称	功　　能
	SnSTART	各任务区程序启动信号(共 32 区)	设置各任务区程序启动信号地址
设置	参见下图		



6. 各任务区停止信号

各任务区停止信号如表 18-7 所示。

表 18-7　各任务区停止信号

类型	参数符号	参数名称	功　　能
	SnSTOP	各任务区程序停止信号(共 32 区)	设置各任务区程序停止信号地址
设置	见下图		



7. 各机器人伺服 ON/OFF

各机器人伺服 ON/OFF 如表 18-8 所示。

表 18-8　各机器人伺服 ON/OFF

类型	参数符号	参数名称	功　　能
	SnSRVON	各机器人伺服 ON	设置各机器人伺服 ON
	SnSRVOFF	各机器人伺服 OFF	设置各机器人伺服 OFF
设置	参见下图,N=1～3		

续表

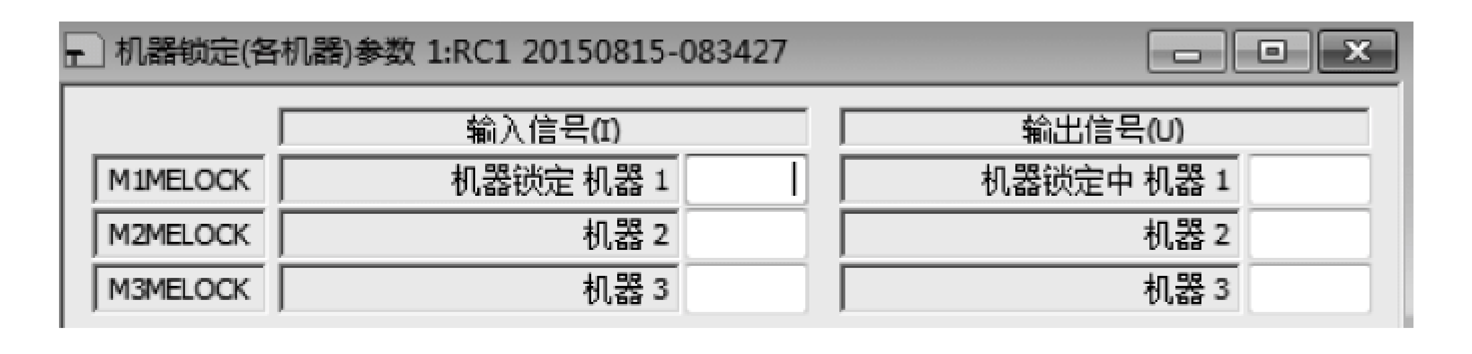


8. 各机器人机械锁定

各机器人机械锁定如表 18-9 所示。

表 18-9　各机器人机械锁定

类型	参数符号	参数名称	功　　能
	SnMELOCK	各机器人机械锁定	设置各机器人机械锁定信号
设置	参见下图,N=1～3		



9. 选择各机器人预热运行模式

各机器人预热运行模式如表 18-10 所示。

表 18-10　各机器人预热运行模式

类型	参数符号	参数名称	功　　能
	MnWUPENA	各机器人预热运行模式选择	设置各机器人预热运行模式
设置	必须预先设置参数 WUPENA,选择预热模式有效。本参数只是对各机器人的选择		



10. 附加轴

附加轴指机器人外围设备中的由伺服系统驱动的运动轴。为了使其配合机器人的动作，可以从机器人控制器一侧对其进行控制，如图 18-5 和表 18-11～表 18-17 所示。

图 18-5 是对附加轴参数进行设置的界面。



图 18-5　附加轴相关参数设置界面

表 18-11　AXMENO——控制附加轴的“机器人号”

类型	参数符号	参数名称	功　　能
	AXMENO	控制附加轴的“机器人号”	设置控制附加轴的机器人编号
设置	如下图		



表 18-12　AXJNO——附加轴的“驱动器站号”

类型	参数符号	参数名称	功　　能
	AXJNO	附加轴的“驱动器站号”	设置附加轴的驱动器站号
设置	如下图，在附加轴连接完毕后，要设置每一驱动器的“站号”。在通用伺服系统中也是要设置站号的		



表 18-13　AXUNT——附加轴使用的单位(deg/mm)

类型	参数符号	参数名称	功　　能
	AXUNT	附加轴使用单位(deg/mm)	设置附加轴使用单位(deg/mm)
设置	如下图，设置附加轴使用单位 AXUNT＝0，deg；AXUNT＝1，mm		



表 18-14　AXSPOL——附加轴旋转方向

类型	参数符号	参数名称	功　　能
	AXSPOL	附加轴旋转方向	确定附加轴旋转方向
设置	如下图。AXSPOL＝0，CCW；AXSPOL＝0，CW		

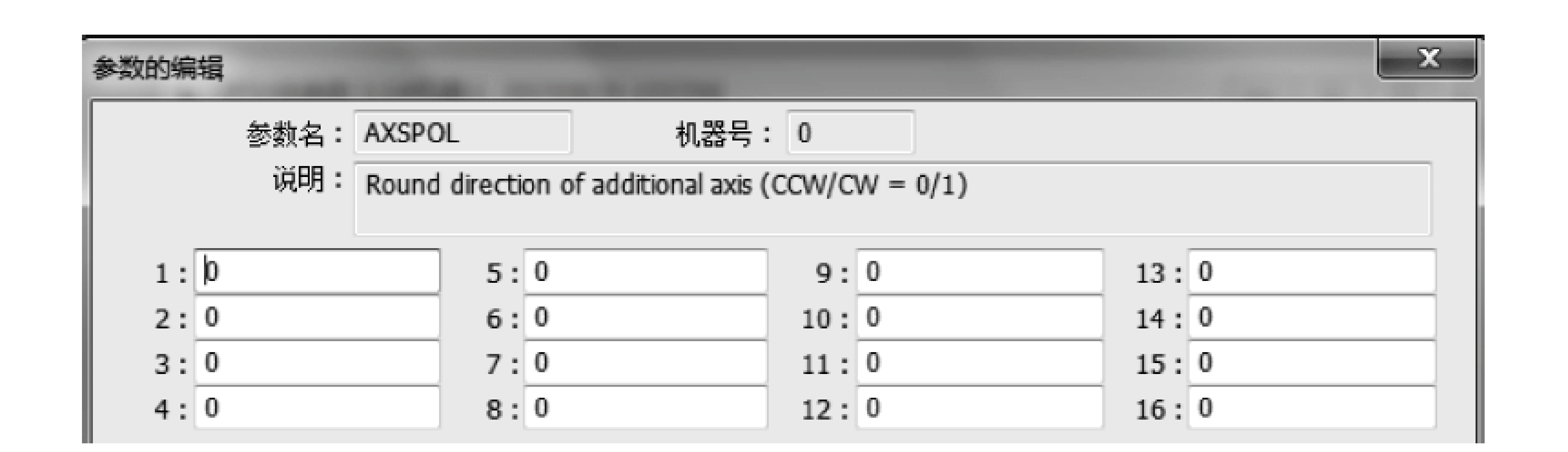


表 18-15　AXACC——附加轴加速时间

类型	参数符号	参数名称	功　　能
	AXACC	附加轴加速时间	设置附加轴加速时间
设置	如下图，设置单位为 sec		



表 18-16　AXDEC——附加轴减速时间

类型	参数符号	参数名称	功　　能
	AXDEC	附加轴减速时间	设置附加轴减速时间
设置	如下图，设置单位为 sec		



表 18-17　AXGRTN——附加轴齿轮比分子

类型	参数符号	参数名称	功　　能
	AXGRTN	附加轴齿轮比分子	设置附加轴齿轮比分子
设置	如下图		



18.2　如何监视输入输出信号

1. 通用信号的监视和强制输入输出

功能：用于监视输入输出信号的 ON/OFF 状态。

单击“监视”→“信号监视”→“通用信号”，弹出“通用信号”框，如图 18-6 所示。在“通用信号”框内除了监视当前输入输出信号的 ON/OFF 状态以外，还可以：

（1）模拟输入信号；

（2）设置监视信号的范围；

（3）强制输出信号 ON/OFF。

2. 对已经命名的输入输出信号监视

功能：用于监视已经命名的输入输出信号的 ON/OFF 状态。

单击“监视”→“信号监视”→“带名字的信号”，弹出“带名字的信号”框，如图 18-7 所示。在“带名字的信号”框内可以监视已经命名的输入输出信号的 ON/OFF 状态。

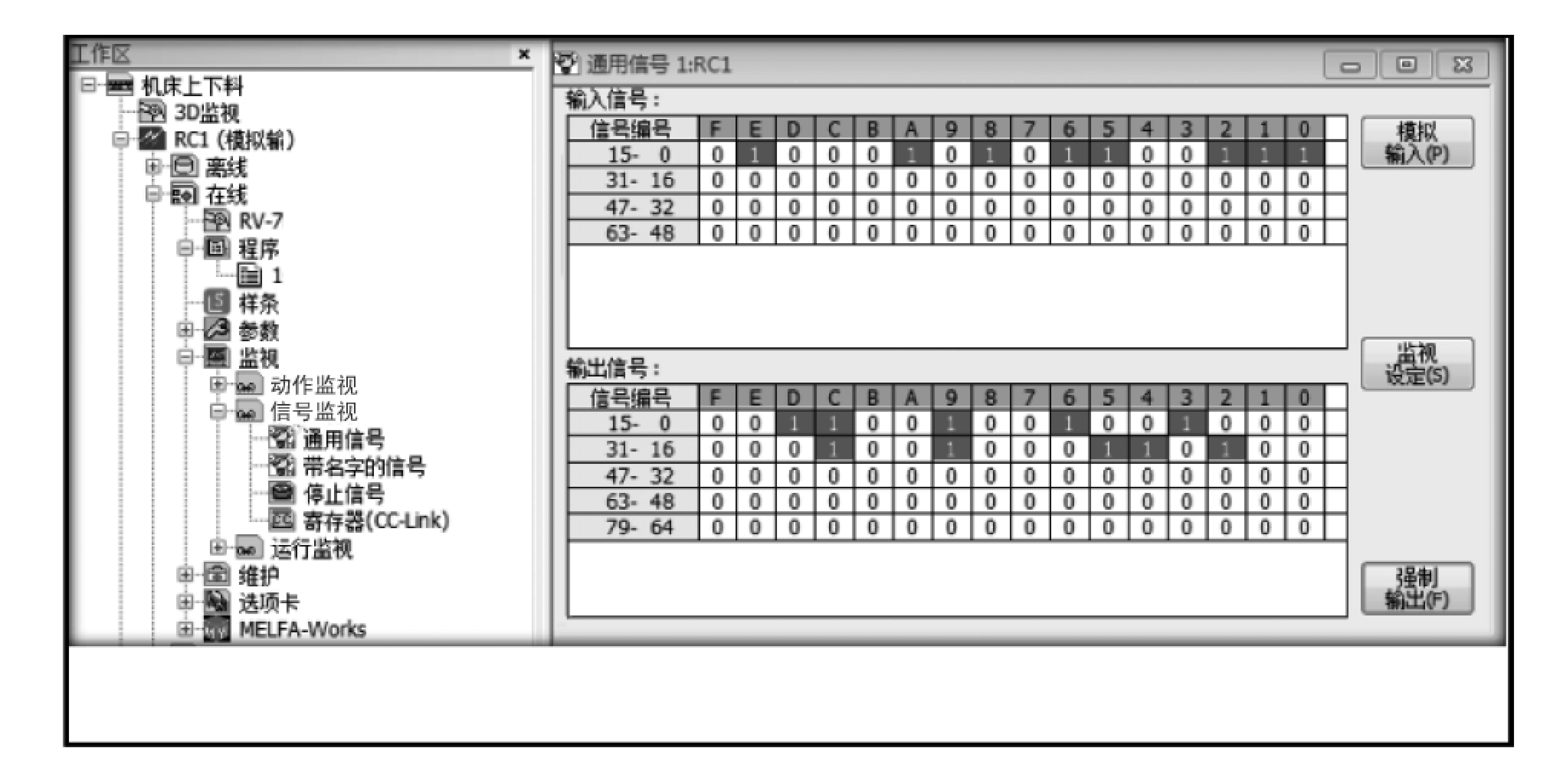


图 18-6 “通用信号”框的监视状态



图 18-7 “带名字的信号”框内监视已经命名的输入输出信号的 ON/OFF 状态

18.3 需要思考的问题

(1) 机器人系统是否具备内置的输入输出功能?

(2) 外接 I/O 卡上的端子未加定义之前,是不是空白的 I/O 端子?

(3) 如何定义各输入输出端子的功能?

(4) 什么是单循环运行? 如何设置单循环运行信号?

(5) 可以通过外部 I/O 端子实现 JOG 运行吗?

(6) 什么是机械锁定功能? 如何设置机械锁定功能?

第19章 第19日——使用外部信号选择程序

【学习目的】

机器人系统内可预先存放很多程序，要运行某一个程序，可以进行选择。选择程序的方法有多种，本章结合第18章的内容，学习仅使用外部端子实现选择程序的方法。

19.1 第一种方法：先选择程序号再启动程序

操作步骤如下。

1. 相关参数设置

(1) 设置参数 PST ＝0，如图 19-1 所示。

PST 是程序选择模式。

PST＝0，先选择程序号再启动；

PST＝1，同时发出“选择程序”与“程序启动”信号。

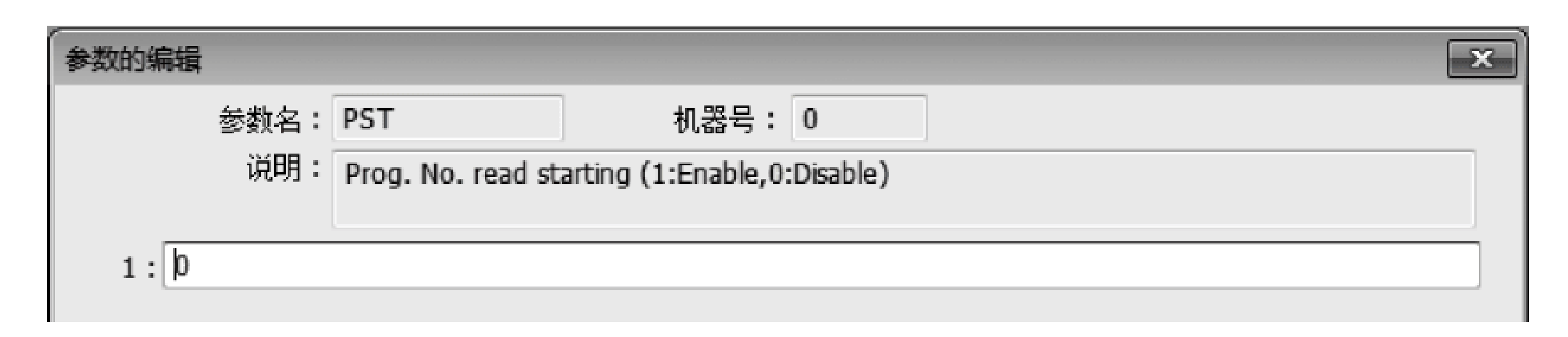


图 19-1 设置参数 PST

(2) 需要处理分配下列输入输出信号。

将输入输出功能分配到下列端子，如表 19-1 所示。

表 19-1 需要使用的输入输出功能及端子分配

参数	对应输入输出信号	功 能	输入端子	输出端子
IOENA	操作权	设置外部 I/O 信号有效	5	3
PRGOUT	输出	将任务区内的“程序号”输出到外部端子，用于检查是否与选择的“程序号”相符	7	
IODATA	数据输入端子范围	设置用以输入数据的端子“起始号”及“结束号”这些端子构成的 8421 码即选择的“程序号”	8～11	8～11
PRGSEL	用于确定“已经选择的程序号”		6	
START	启动		3	

将以上参数功能分配到对应的“输入信号端子”。

(3) 在 RT ToolBox 软件上的具体设置如图 19-2 和图 19-3 所示。



图 19-2　设置输入输出信号端子

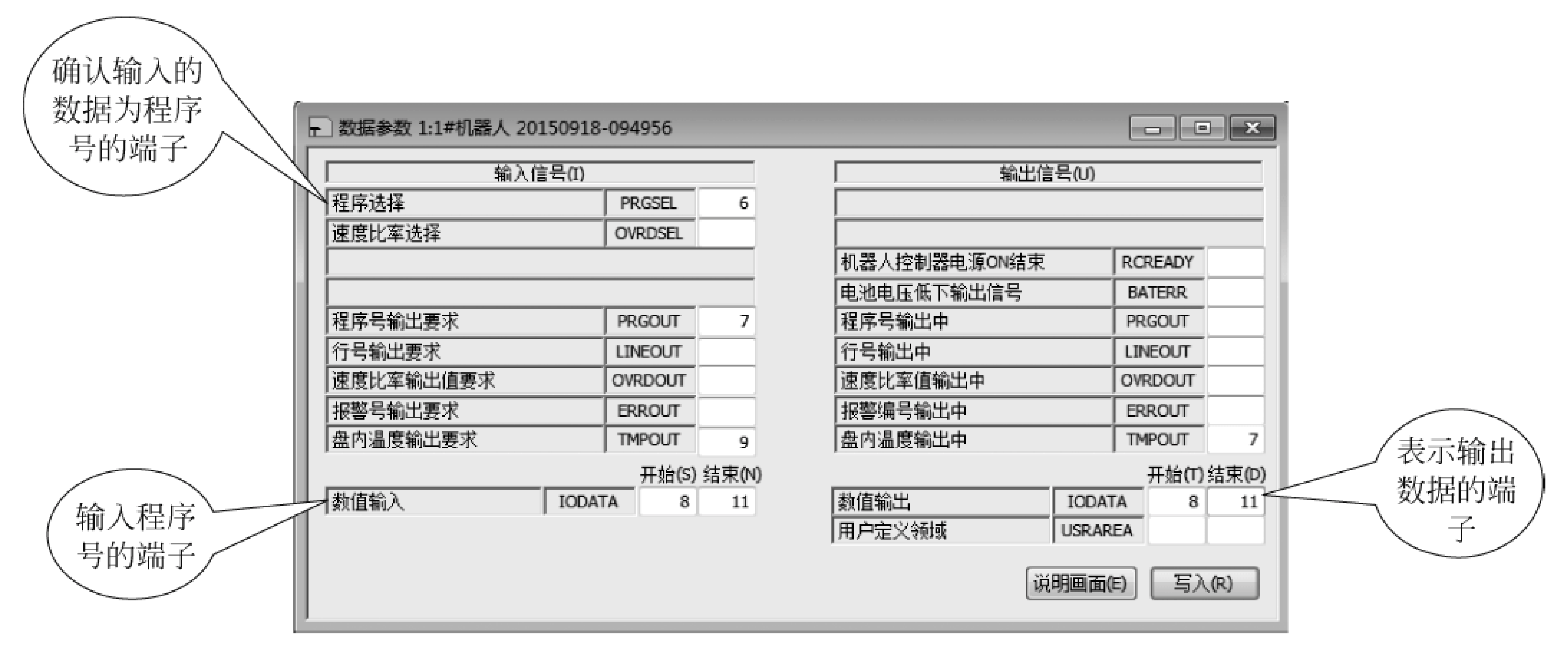


图 19-3　设置输入输出信号端子

2. 操作

(1) 指令 IOENA=1(输入信号 5=ON),使外部操作信号有效。

(2) 选择程序号:以端子 8～11 构成的 8421 码选择程序号。示例如表 19-2～表 19-4 所示。

表 19-2　选择 3 号程序

端子 11	端子 10	端子 9	端子 8
0	0	1	1

表 19-3　选择 7 号程序

端子 11	端子 10	端子 9	端子 8
0	1	1	1

表 19-4　选择 12 号程序

端子 11	端子 10	端子 9	端子 8
1	1	0	0

3. 确认已经选择的程序号有效

(1) 操作 PRESEL 端子(输入端子 6)=ON,其功能是确认已经选择的程序号生效。

(2) 操作 PRGOUT 端子(输入端子 7)=ON。

观察输出端子 8～11 构成的程序号是否与选定的程序号相同,如果相同可以执行“启动”。

4. 发出“启动”信号

启动已经选择的程序。

操作信号时序图如图 19-4 所示。

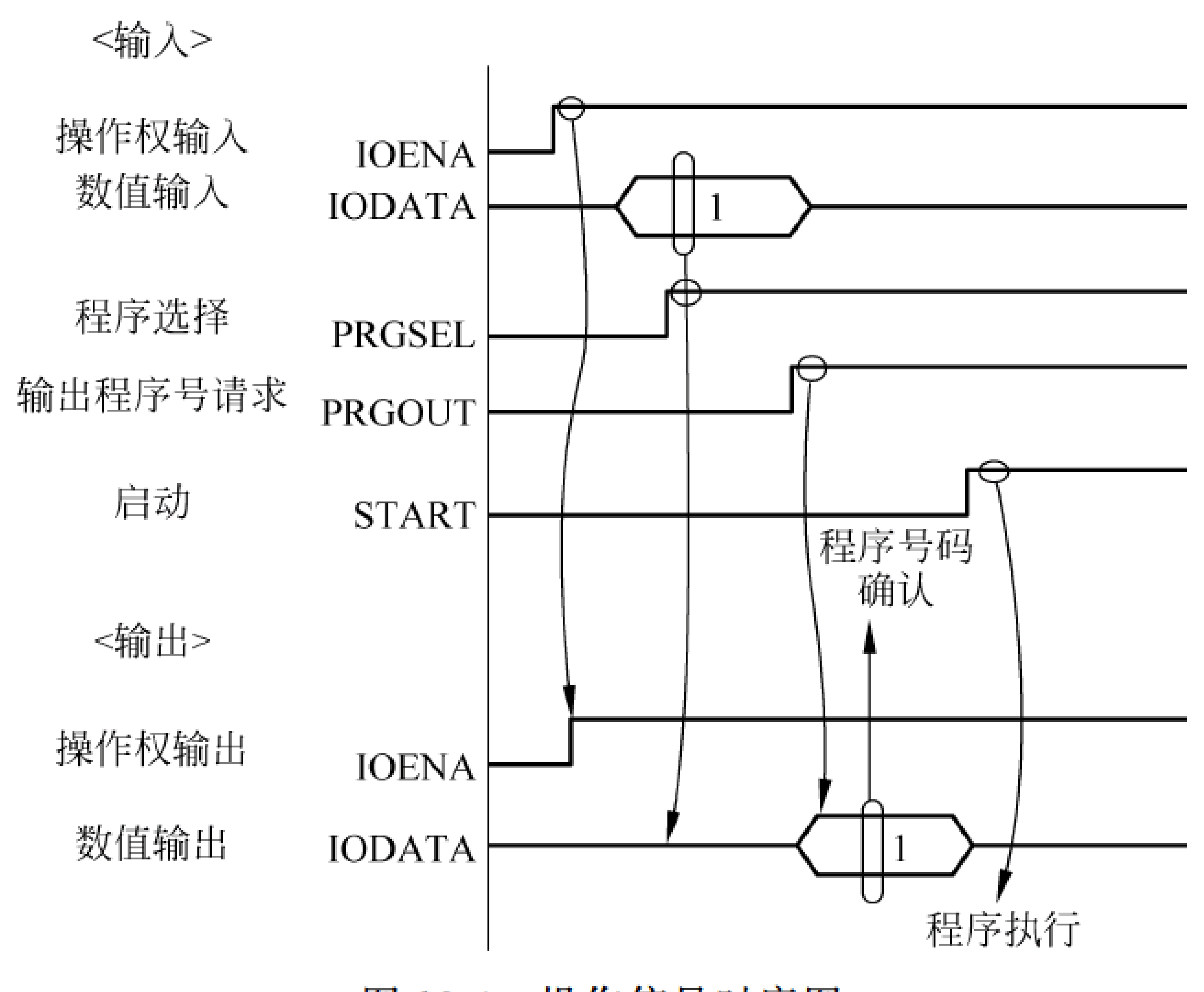


图 19-4　操作信号时序图

19.2　第二种方法:“选择程序号”与“启动”信号同时生效

这种方法比较快捷,但是必须在保证程序号正确的情况下进行。

操作步骤如下。

(1) 设置参数 PST =1。

PST=1:“选择程序”与“程序启动”同时生效。

(2) 操作:指令 IOENA=1(输入信号 5=ON)。

使外部操作信号有效。

(3) 选择程序:以端子 8～11 构成的 8421 码选择程序号。示例如表 19-5 所示。

表 19-5　选择 12 号程序

端子 11	端子 10	端子 9	端子 8
1	1	0	0

（4）发出“启动”信号

启动已经选择的程序。

操作信号时序图如图 19-5 所示。

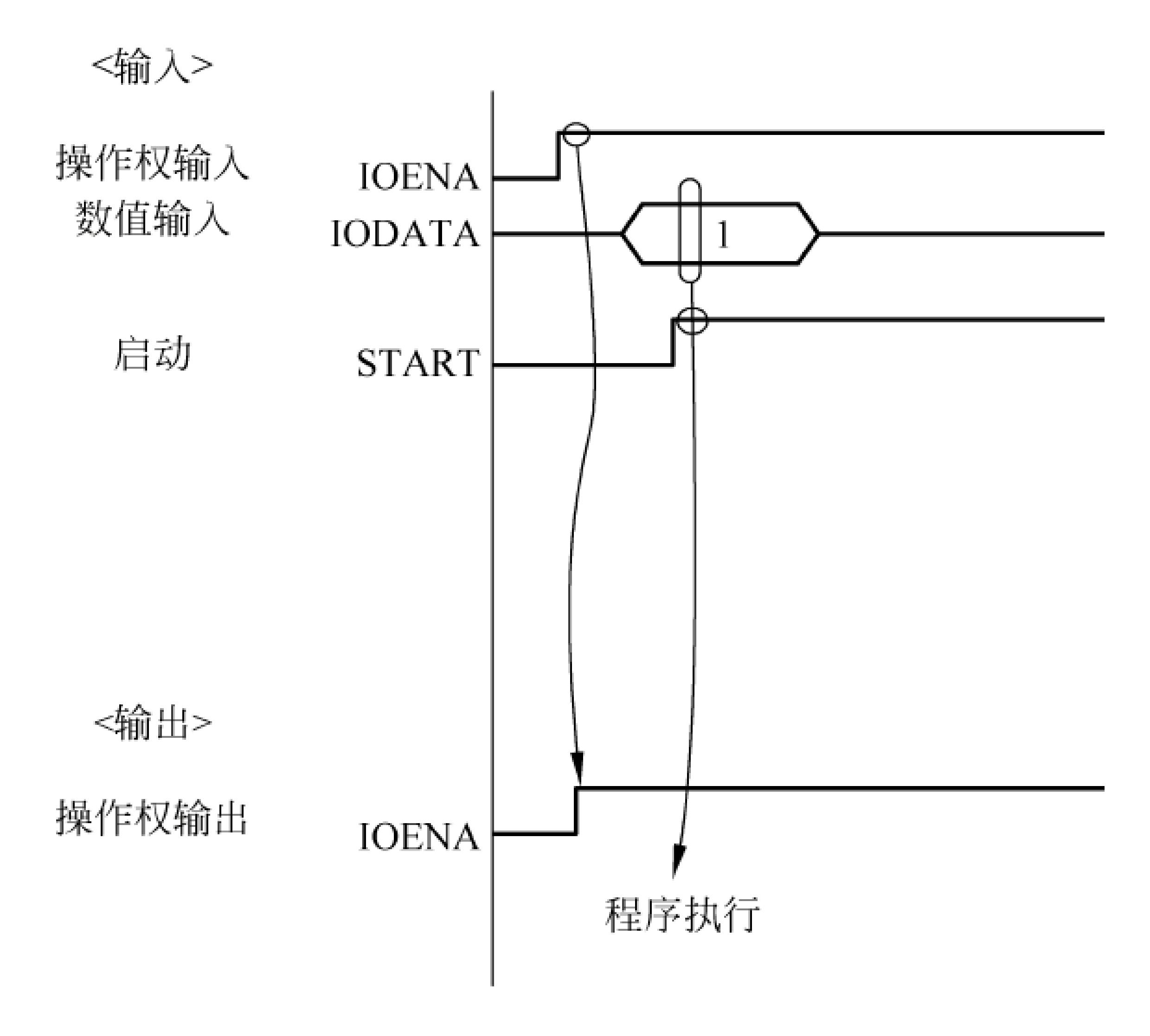


图 19-5　操作信号时序图

19.3　需要思考的问题

（1）机器人系统可以预置多少工作程序？

（2）如何定义输入数据端子？输入数据端子如何构成数据？

（3）如何定义输出数据端子？怎样读取输出数据端子构成的数据？

（4）怎样确认输入的数据就是程序号？

（5）哪种选择程序号的方法更安全？

第 20 章　第 20 日——编程语言中的函数

【学习目的】

在机器人的编程言语中，提供了大量的运算函数，这样就大大提高了编程的便利性。本章将详细介绍这些运算函数的用法，这些运算函数按英文字母顺序排列，便于学习和查阅。在学习本章时，应该先通读一遍，然后根据编程需要，重点研读需要使用的函数。

20.1　Abs——求绝对值

1. 功能

Abs 为求绝对值函数。

2. 格式

```
<数值变量> = Abs <数式>
```

3. 例句

```
P2.C = Abs(P1.C) '——将 P1 点 C 轴数据求绝对值后赋予 P2 点 C 轴。
Mov P2 '——前进到 P2 点。
M2 =- 100 '——赋值。
M1 = Abs(M2) '——将 M2 求绝对值后赋值到 M1。
```

20.2　Asc——求字符串的 ASCII 码

1. 功能

Asc 用于求字符串的 ASCII 码。

2. 格式

```
<数值变量> = Asc <字符串>
```

3. 例句

```
M1 = Asc("A") '——M1  = &H41
```

20.3　Atn/Atn2——余切函数

1. 功能

Atn/Atn2 为余切函数，用于计算余切。

2. 格式

```
<数值变量> = Atn<数式>
<数值变量> = Atn2<数式1>,<数式2>
```

3. 术语解释

```
<数式>——ΔY/ΔX
<数式1>——ΔY
<数式2>——ΔX
```

4. 例句

```
M1 = Atn(100/100)'——M1 = π/4 弧度
M2 = Atn2(-100,100)'—— M1 =- π/4 弧度
```

5. 说明

根据数据计算余切，单位为“弧度”。

Atn 范围在$-\pi/2\sim\pi/2$。

Atn2 范围在$-\pi\sim\pi$。

20.4 CalArc——计算圆弧数据

1. 功能

CalArc 用于当指定的三点构成一段圆弧时，求出圆弧的半径、中心角和圆弧长度。

2. 格式

```
<数值变量4> = CalArc(<位置1>,<位置2>,<位置3>,<数值变量1>,<数值变量2>,<数值变量3>,
<位置变量1>)
```

3. 术语解释

<位置1>——圆弧起点；

<位置2>——圆弧通过点；

<位置3>——圆弧终点；

<数值变量1>——计算得到的“圆弧半径(mm)”；

<数值变量2>——计算得到的“圆弧中心角(deg)”；

<数值变量3>——计算得到的“圆弧长度(mm)”；

<位置变量1>——计算得到的“圆弧中心坐标(位置型，ABC=0)；

<数值变量4>——函数计算值：

<数值变量4>=1 可正常计算；

<数值变量4>=-1——给定的两点为同一点，或三点在同一直线上；

<数值变量4>=-2——给定的三点为同一点。

4. 例句

```
M1 = CalArc(P1,P2,P3,M10,M20,M30,P10)'——做求圆弧各参数计算。
If M1 <> 1 Then End'——如果各设定条件不对，就结束程序。
```

```
MR = M10'——将"圆弧半径"代入 MR。
MRD = M20'——将"圆弧中心角"代入 MRD。
MARCLEN = M30'——将"圆弧长度"代入 MARCLEN。
PC = P10'——将"圆弧中心点坐标"代入 PC。
```

20.5 CInt——将数据四舍五入后取整

1. 功能

CInt 用于将数据四舍五入后取整。

2. 格式

```
<数值变量> = CInt(<数据>)
```

3. 例句

```
M1 = CInt(1.5)'—— M1 = 2
M2 = CInt(1.4)'—— M2 = 1
M3 = CInt( - 1.4)'—— M3 =- 1
M4 = CInt( - 1.5)'—— M4 =- 2
```

20.6 Cos——余弦函数

1. 功能

Cos 为余弦函数。

2. 格式

```
<数值变量> = Cos(<数据>)
```

3. 例句

```
M1 = Cos(Rad(60))'——将 60°的余弦值代入 M1。
```

4. 说明

(1) 角度单位为“弧度”。

(2) 计算结果范围－1～1。

20.7 Deg——将角度单位从弧度(rad)变换为度(deg)

1. 功能

Deg 用于将角度单位从弧度(rad)变换为度(deg)。

2. 格式

```
<数值变量> = Deg(<数式>)
```

3. 例句

```
1  P1 = P_Curr'——设置 P1 为"当前值"。
2  If Deg(P1.C)< 170 Or Deg(P1.C)> - 150 Then * NOErr1'——如果 P1.C 的度数(deg)小于 170°或大于 - 150°,则跳转到 * NOErr1。
3  Error 9100'——报警。
4   * NOErr1'——程序分支标志。
```

20.8　Dist——求两点之间的距离

1. 功能

求两点之间的距离(mm)。

2. 格式

<数值变量> = Dist(<位置 1 >,<位置 2 >)

3. 例句

```
1  M1 = Dist(P1,P2)'—— M1 为 P1 与 P2 点之间的距离。
```

4. 说明

J 关节点无法使用本功能。

20.9　Exp——计算以 e 为底的指数函数

1. 功能

计算以 e 为底的指数函数。

2. 格式

<数值变量> = Exp(<数式>)

3. 例句

```
1  M1 = Exp(2)'—— M1  = e^2。
```

20.10　Fix——计算数据的整数部分

1. 功能

计算数据的整数部分。

2. 格式

<数值变量> = Fix(<数式>)

3. 例句

```
1  M1 = Fix(5.5)'—— M1  = 5。
```

20.11　Fram——建立坐标系

1. 功能

由给定的三个点构建一个坐标系标准点。常用于建立新的工件坐标系。

2. 格式

```
<位置变量 4 > = Fram(<位置变量 1 >,<位置变量 2 >,<位置变量 3 >)
```

3. 术语解释

<位置变量 1 >：新平面上的原点。

<位置变量 2 >：新平面上的 X 轴上的一点。

<位置变量 3 >：新平面上的 Y 轴上的一点。

<位置变量 4 >：新坐标系基准点。

4. 例句

```
Base P_NBase'——初始坐标系
P10 = Fram(P1,P2,P3)'——求新建坐标系(P1,P2,P3)原点 P10 在世界坐标系中的位置。
P10 = Inv(P10)'——转换。
Base P10'——Base P10 为新建世界坐标系。
```

20.12　Int——计算数据最大值的整数

1. 功能

Int 用于计算数据最大值的整数。

2. 格式

```
<数值变量> = Int(<数式>)
```

3. 例句

```
M1 = Int(3.3)'—— M1 = 3。
```

20.13　Inv——对位置数据进行“反向变换”

1. 功能

Inv 用于对位置数据进行“反向变换”。

Inv 指令可用于根据当前点建立新的“工件坐标系”，如图 20-1 所示。在视觉功能中，也可以用于计算偏差量。

2. 格式

```
<位置变量> = Inv <位置变量>
```

3. 例句

```
1  P1 = Inv(P1) '——对位置数据 P1 进行"反向变换"。
```

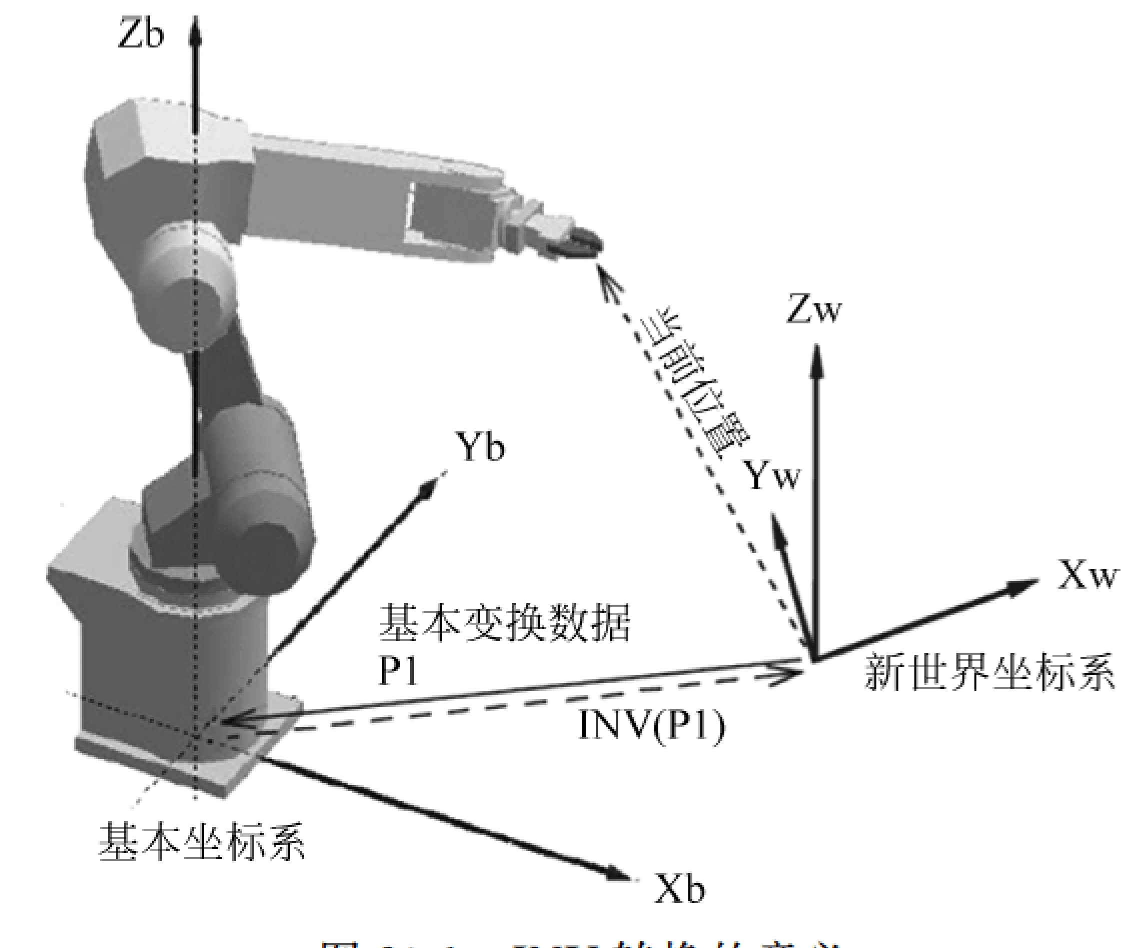


图 20-1　INV 转换的意义

4. 说明

(1) 在原坐标系中确定一点 P1；

(2) 如果希望以 P1 点作为新坐标系的原点，则使用指令 INV 进行变换，即 P1＝INV(P1)，则以 P1 为原点建立了新的坐标系。注意图中 INV(P1)的效果。

20.14　JtoP——将关节位置数据转成"直角坐标系数据"

1. 功能

JtoP 用于将关节位置数据转成"直角坐标系数据"。

2. 格式

```
<位置变量> = JtoP <关节变量>
```

3. 例句

```
1  P1 = JtoP(J1) '——将关节数据 J1 转成直角坐标系数据 P1。
```

4. 说明

注意：J1 为关节变量；P1 为位置型变量。

20.15　Log——计算常用对数

1. 功能

Log 用于计算常用对数(以 10 为底的对数)。

2. 格式

```
<数值变量> = Log <数式>
```

3. 例句

```
M1 = Log (2)'—— M1 = 0.301030。
```

20.16 Max——计算最大值

1. 功能

Max 用于求出一组数据中的最大值。

2. 格式

```
<数值变量> = Max(<数式 1 >,<数式 2 >,<数式 3 >)
```

3. 例句

```
M1 = Max(2,1,3,4,10,100)'—— M1 = 100。
```

这一组数据中最大的数是 100。

20.17 Min——求最小值

1. 功能

Min 用于求出一组数据中的最小值。

2. 格式

```
<数值变量> = Min(<数式 1 >,<数式 2 >,<数式 3 >)
```

3. 例句

```
M1 = Min (2,1,3,4,10,100)'——M1 = 1。
```

这一组数据中最小的数是 1。

20.18 PosCq——检查给出的位置点是否在允许动作区域内

1. 功能

PosCq 用于检查给出的位置点是否在允许动作范围区域内。

2. 格式

```
<数值变量> = PosCq <位置变量>
```

<位置变量>：可以是直交型也可以是关节型位置变量。

3. 例句

```
M1 = PosCq(P1)'——检查 P1 点是否在允许动作范围区域内。
```

4. 说明

如果 P1 点在动作范围以内，M1＝1。

如果 P1 点在动作范围以外，M1＝0。

20.19 PosMid——求出两点之间做直线插补的中间位置点

1. 功能

PosMid 用于求出两点之间做直线插补的中间位置点。

2. 格式

```
<位置变量> = PosCq <位置变量 1 >,<位置变量 2 >,<数式 1 >,<数式 2 >
```

<位置变量 1 >：直线插补起点。
<位置变量 2 >：直线插补终点。

3. 例句

```
1  P1 = PosMid(P2,P3,0,0)'—— P1 点为 P2、P3 点的中间位置点。
```

20.20 PtoJ——将直交型位置数据转换为关节型数据

1. 功能

PtoJ 用于将直交型位置数据转换为关节型数据。

2. 格式

```
<关节位置变量> = PtoJ <直交位置变量>
```

3. 例句

```
1  J1 = PtoJ(P1)'——将直交型位置数据 P1 转换为关节型数据 J1。
```

4. 说明

J1 为关节型位置变量。P1 为直交型位置变量。

20.21 Rad——将角度单位转换为弧度单位

1. 功能

Rad 用于将角度(deg)单位转换为弧度单位(rad)。

2. 格式

```
<数值变量> = PtoJ <数式>
```

3. 例句

```
1  P1 = P_Curr'——设置 P1 为当前位置。
2  P1.C = Rad(90)'——将 P1 的 C 轴数值转换为弧度。
3  Mov P1'——前进到 P1 点。
```

4. 说明

常常用于对位置变量中形位(A/B/C)的计算和三角函数的计算。

20.22　Rdfl2——求指定关节轴的“旋转圈数”

1. 功能

Rdfl2 用于求指定关节轴的“旋转圈数”，即求结构标志 FL2 的数据。

2. 格式

```
<设置变量> = Rdfl2(<位置变量>,<数式>)
```

<数式>：指定关节轴。

3. 例句

```
P1 = (100,0,100,180,0,180)(7,&H00100000)'——设置 P1 点。
M1 = Rdfl2(P1,6)'——将 P1 点 C 轴"旋转圈数"赋值到 M1。
```

4. 说明

(1) 取得的数据范围：－8～7。

(2) 结构标志 FL2 由 32b 构成。旋转圈数为－1～－8 时，显示形式为 F～8。

在 FL2 标志中，FL2＝00000000，bit0～7 对应轴号 87654321。每 1 位的数值代表旋转的圈数。正数表示正向旋转的圈数。

例如：

J6 轴旋转圈数＝＋1 圈，则 FL2＝00100000。

J6 轴旋转圈数＝－1 圈，则 FL2＝00F00000。

20.23　Rnd——产生一个随机数

1. 功能

Rnd 用于产生一个随机数。

2. 格式

```
<数值变量> = Rnd(<数式>)
```

<数式>：指定随机数的初始值。

<数值变量>：数据范围 0.0～1.0。

3. 例句

```
Dim MRND(10)'——定义数组。
C1 = Right$(C_Time,2)'——(截取字符串)C1 = "me"。
MRNDBS = Cvi(C1)'——将字符串"me"转换为数值。
MRND(1) = Rnd(MRNDBS)'——以 MRNDBS 为初始值产生一个随机数。
For M1 = 2 To 10'——做循环指令及条件。
MRND(M1) = Rnd(0)'——以 0 为初始值产生一个随机数赋值到 MRND(2)～ MRND(10)
Next M1'——进入下一循环。
```

20.24 SetJnt——设置各关节变量的值

1. 功能

SetJnt 用于设置“关节型位置变量”。

2. 格式

<关节型位置变量>= SetJnt <J1 轴>,<J2 轴>,<J3 轴>,<J4 轴>,<J5 轴>,<J6 轴>,<J7 轴>,<J8 轴>

<J1 轴>、<J2 轴>：单位为弧度(rad)。

3. 例句

```
1  J1 = J_Curr'——设置 J1 为当前值。
2  For M1 = 0 To 60 Step 10'——设置循环指令及循环条件。
3  M2 = J1.J3 + Rad(M1)'——将 J1 点 J3 轴数据加 M1(弧度值)后赋值到 M2。
4  J2 = SetJnt(J1.J1, J1.J2, M2)'——设置使 J2 点数据(其中 J3 轴每次增加 10°,J4 轴以后为相同的值)。
5  Mov J2'——前进到 J2 点。
6  Next M1'——下一循环。
7  M0 = Rad(0)'——取弧度值。
8  M90 = Rad(90)'——取弧度值。
9  J3 = SetJnt(M0,M0,M90,M0,M90,M0)'——设置 J3 点。
10 Mov J3'——前进到 J3 点。
```

20.25 SetPos——设置直交型位置变量数值

1. 功能

设置直交型位置变量数值。

2. 格式

<位置变量>= SetPos <X 轴>,<Y 轴>,<Z 轴>,<A 轴>,<B 轴>,<C 轴>,<L1 轴>,<L2 轴>

3. 术语解释

<X 轴>~<Z 轴>：单位为 mm。

<A 轴>~<C 轴>：单位为弧度(rad)。

4. 例句

```
1  P1 = P_Curr'——设置 P1 为当前值。
2  For M1 = 0 To 100 Step 10'——设置循环指令及循环条件。
3  M2 = P1.Z + M1'——将"P1.Z + M1"赋值到"M2"。
4  P2 = SetPos(P1.X, P1.Y, M2)'——设置 P2 点(Z 轴数值每次增加 10mm。A 轴以后各轴数值不变)。
5  Mov P2'——前进到 P2 点。
6  Next M1'——下一循环。
```

20.26 Sgn——求数据的符号

1. 功能

求数据的符号。

2. 格式

```
<数值变量> = Sgn <数式>
```

3. 例句

```
1  M1 =- 12 '——赋值。
2  M2 = Sgn(M1) '——求 M1 的符号(M2  =- 1)。
```

4. 说明

<数式>=正数,<数值变量>=1。

<数式>=0,<数值变量>=0。

<数式>=负数,<数值变量>=－1。

20.27 Sin——求正弦值

1. 功能

求正弦值。

2. 格式

```
<数值变量> = Sin <数式>
```

3. 例句

```
1  M1 = Sin(rad(60)) '—— M1 = 0.86603
```

4. 说明

<数式>的单位为弧度。

20.28 Sqr——求平方根

1. 功能

求平方根。

2. 格式

```
<数值变量> = Sqr <数式>
```

3. 例句

```
1  M1 = Sqr(2) '——求 2 的平方根(M1  = 1.41421)
```

20.29 Tan——求正切

1. 功能

求正切。

2. 格式

```
<数值变量> = Tan <数式>
```

3. 例句

```
1  M1 = Tan (rad(60))'—— M1 = 1.73205
```

4. 说明

<数式>的单位为弧度。

20.30 Zone——检查指定的位置点是否进入指定的区域

1. 功能

Zone 用于检查指定的位置点是否进入指定的区域,如图 20-2 所示。

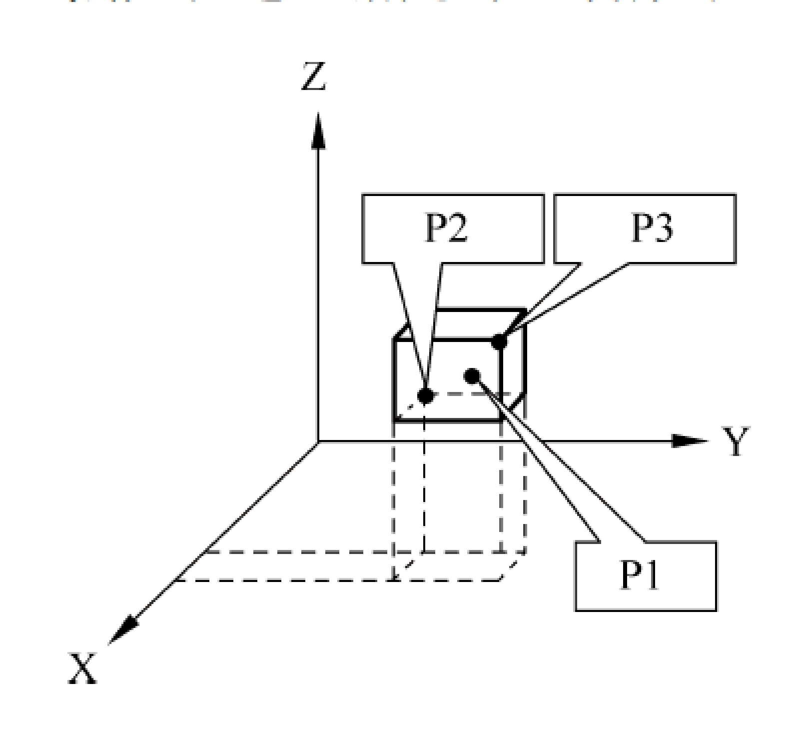


图 20-2 指定的位置点是否进入指定的位置区域

2. 格式

```
<数值变量> = Zone <位置 1 >,<位置 2 >,<位置 3 >
```

<位置 1 >:检测点。

<位置 2 >,<位置 3 >:构成指定区域的空间对角点。

<位置 1 >,<位置 2 >,<位置 3 >为直交型位置点 P。

<数值变量>=1:<位置 1 >点进入指定的区域。

<数值变量>=0:<位置 1 >点没有进入指定的区域。

3. 例句

```
1  M1 = Zone(P1,P2,P3)'——检测 P1 点是否进入指定的空间。
2  If M1 = 1 Then Mov P_Safe Else End'——判断 - 执行语句。
```

20.31　Zone2——检查指定的位置点是否进入指定的区域(圆筒型)

1. 功能

Zone2 用于检查指定的位置点是否进入指定的(圆筒型)区域,如图 20-3 所示。

2. 格式

```
<数值变量> = Zone2 <位置 1 >,<位置 2 >,<位置 3 >,<数式>
```

<位置 1 >：被检测点。

<位置 2 >,<位置 3 >：构成指定圆筒区域的空间点。

<数式>：两端半球的半径。

<位置 1 >,<位置 2 >,<位置 3 >：直交型位置点 P。

<数值变量>=1：<位置 1 >点进入指定的区域。

<数值变量>=0：<位置 1 >点没有进入指定的区域。

Zone2 只用于检查指定的位置点是否进入指定的(圆筒型)区域,不考虑形位。

3. 例句

```
1 M1 = Zone2(P1,P2,P3,50)'——检测 P1 点是否进入指定的空间。
2 If M1 = 1 Then Mov P_Safe Else End'——判断 - 执行语句。
```

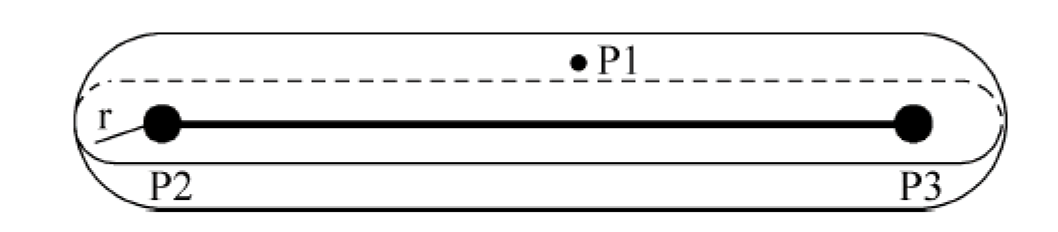


图 20-3　指定的位置点是否进入指定的位置区域

20.32　Zone3——检查指定的位置点是否进入指定的区域(长方体)

1. 功能

检查指定的位置点是否进入指定的区域(长方体)。

2. 格式

```
<数值变量> = Zone <位置 1 >,<位置 2 >,<位置 3 >,<位置 4 >,<数式 W >,<数式 H >,<数式 L >
```

<位置 1 >：检测点。

<位置 2 >,<位置 3 >：构成指定区域的空间点。

<位置 4 >：与<位置 2 >、<位置 3 >共同构成指定平面的点。

<位置 1 >,<位置 2 >,<位置 3 >为直交型位置点 P。

<数式 W >：指定区域宽。

<数式 H >：指定区域高。

<数式 L >：(以<位置 2 >,<位置 3 >为基准)指定区域长。

<数值变量>=1：<位置 1>点进入指定的区域。

<数值变量>=0：<位置 1>点没有进入指定的区域。

3. 例句(如图 20-4)

```
M1 = Zone3(P1,P2,P3,P4,100,100,50)'——检测 P1 点是否进入指定的空间。
If M1 = 1 Then Mov P_Safe Else End'——判断 - 执行语句。
```

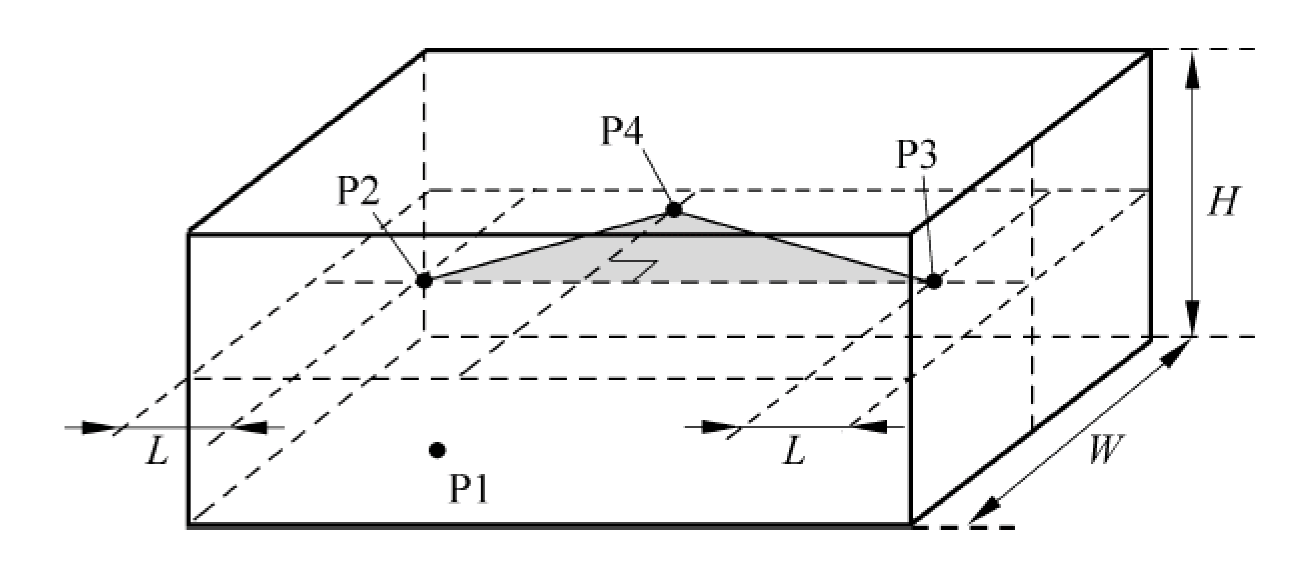


图 20-4　指定的位置点是否进入指定的位置区域

20.33　需要思考的问题

(1) 如何计算两点之间的距离?

(2) 如何对位置数据进行反向变换?

(3) 如何将关节位置数据转成直角坐标系数据?

(4) 如何将直角坐标系数据转成关节位置数据?

(5) 如何使用 Fram 函数构建一个新的坐标系?

第 21 章 第 21 日——机器人在码垛项目中的应用

【学习目的】

本章通过一个实际案例，学习机器人在码垛项目中的应用。在学习本章的内容前，应该先复习第 10 章的内容。

21.1 项目综述

某项目需要使用机器人对包装箱进行码垛处理。如图 21-1 所示，由传送线将包装箱传送到固定位置，再由机器人抓取并码垛。码垛规格要求为 6×8，错层布置，层数＝10，左右各一垛。

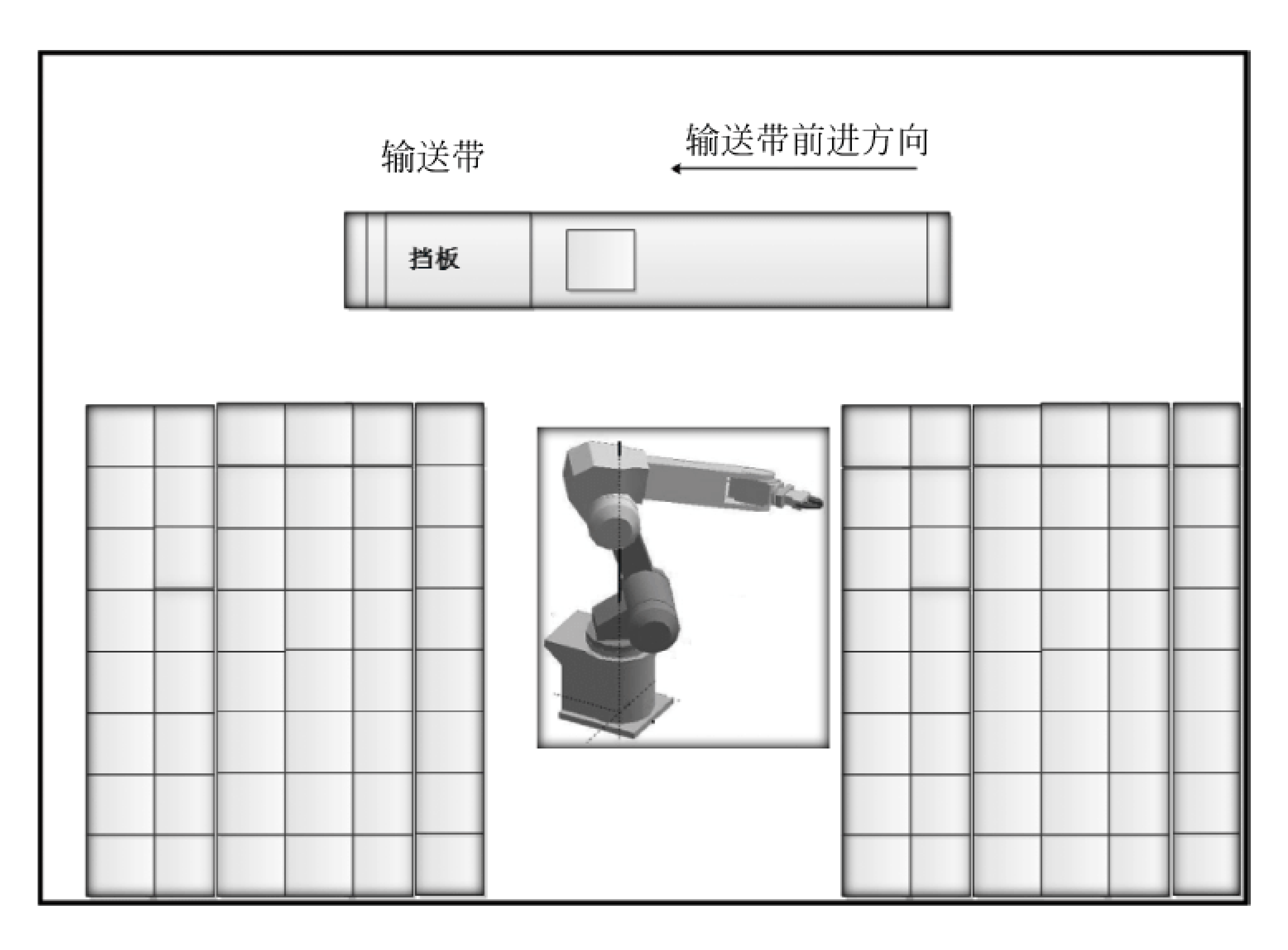


图 21-1 机器人码垛流水线工作示意图

21.2 解决方案

(1) 配置机器人一台作为工作中心，负责工件抓取搬运码垛。机器人配置 32 点输入 32 点输出的 I/O 卡。选取三菱 RV-7FLL 机器人，该机器人搬运重量为 7kg，最大动作半径为

1503mm。由于是码垛作业，所以选取机器人的动作半径要求尽可能大一些。三菱RV-7FLL臂长加长型的机器人，可以满足工作要求。

(2) 示教单元：R33TB(必须选配，用于示教位置点)。

(3) 机器人选件：输入输出信号卡2D-TZ368(用于接收外部操作屏信号和控制外围设备动作)。

(4) 选用三菱PLC FX3U-48MR作主控系统，用于控制机器人的动作并处理外部检测信号。

(5) 触摸屏选用GS2110。触摸屏可以直接与机器人相连接，直接设置和修改各工艺参数，发出操作信号。

21.2.1 硬件配置

根据技术经济性分析，选定硬件配置如表21-1所示。

表21-1 硬件配置一览表

序号	名称	型号	数量	备注
1	机器人	RV-7FLL	1	三菱
2	简易示教单元	R33TB	1	三菱
3	输入输出卡	2D-TZ368	1	三菱
4	PLC	FX3U-48MR	1	三菱
5	GOT	GS2110-WTBD	1	三菱

21.2.2 输入输出点分配

根据现场控制和操作的需要，设计输入输出点，输入输出点通过机器人I/O卡TZ-368接入，TZ-368的地址编号是机器人识别的I/O地址。为识别方便，分列输入输出信号。输入信号一览表如表21-2所示，输出信号一览表如表21-3所示。

表21-2 输入信号一览表

序号	输入信号名称	输入地址(TZ-368)
1	自动启动	3
2	自动暂停	0
3	复位	2
4	伺服ON	4
5	伺服OFF	5
6	报警复位	6
7	操作权	7
8	回退避点	8
9	机械锁定	9
10	气压检测	10
11	输送带正常运行检测	11
12	输送带进料端有料无料检测	12

续表

序　　号	输入信号名称	输入地址(TZ-368)
13	输送带无料时间超长检测	13
14	1 垛位有料无料检测	14
15	2 垛位有料无料检测	15
16	吸盘夹紧到位检测	29
17	吸盘松开到位检测	30

表 21-3　输出信号一览表

序　　号	输出信号名称	输出信号地址(TZ-368)
1	机器人自动运行中	0
2	机器人自动暂停中	4
3	急停中	5
4	报警复位	2
5	吸盘 ON	11
6	吸盘 OFF	12
7	输送带无料时间超长报警	13

21.3　编　　程

21.3.1　总工作流程

总工作流程如图 21-2 所示。

(1) 初始化程序。

(2) 输送带有料无料判断。

① 如果无料,继续判断是否超过无料等待时间,如果超过,则进入报警程序;再跳到程序 END 处。

② 如果未超过“无料等待时间”,则继续进行有料无料判断。

③ 输送带有料无料判断:如果有料,则进行 1#垛可否执行码垛作业判断,如果 Yes,则执行 1#码垛作业。如果 No,则执行 2#码垛作业。

④ 进行 2#垛可否执行码垛作业判断,如果 Yes,则执行 2#码垛作业,如果 No,则跳到报警提示程序,再执行结束 END。

21.3.2　编程计划

1. 程序结构分析

必须从宏观着手编制主程序,只有在编制主程序时考虑周详,无所遗漏,安全可靠,保护严密,才能达到事半功倍的效果。

在总流程图上主程序可以分为 4 个二级程序,如表 21-4 所示。

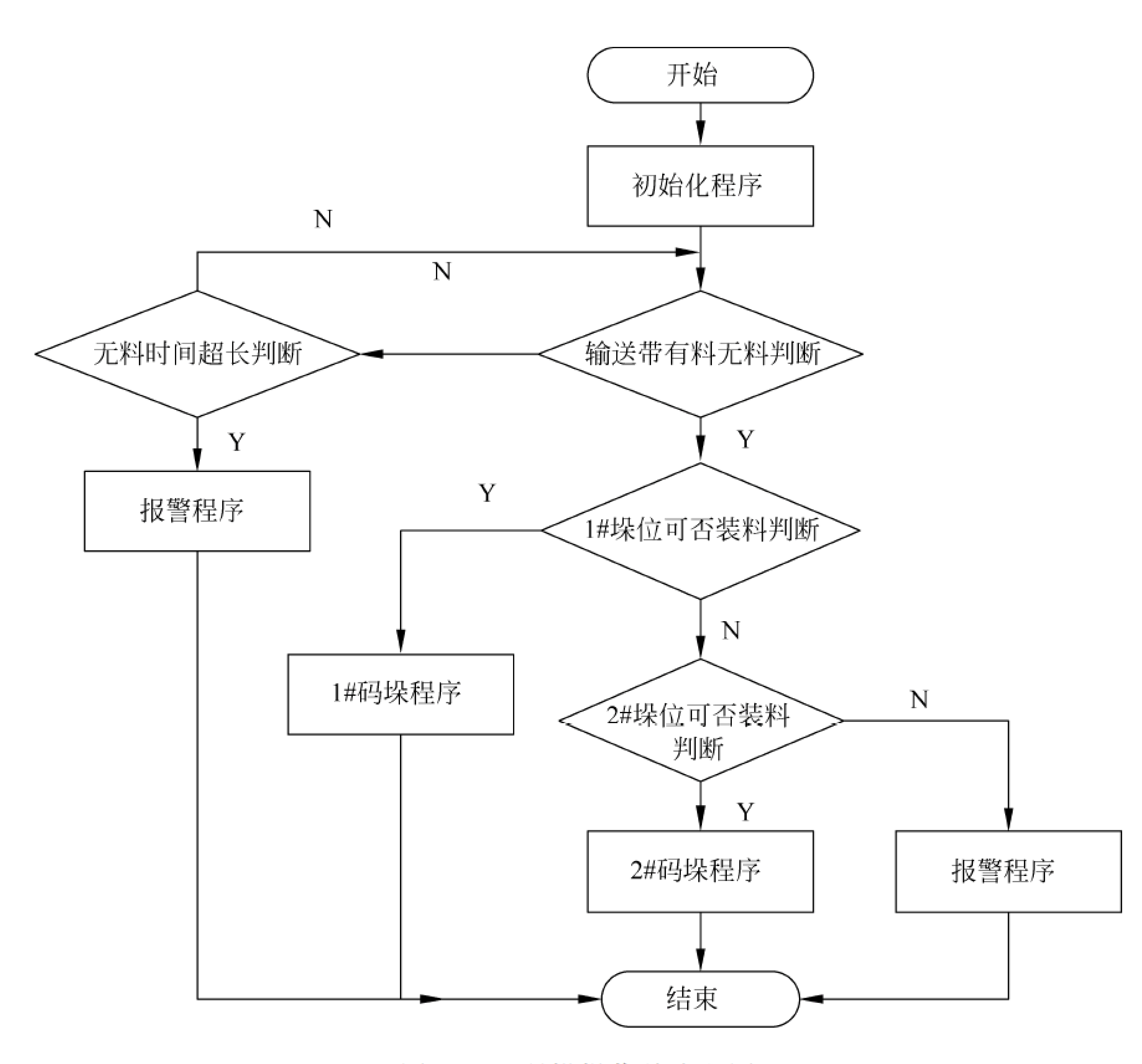


图 21-2　码垛操作总流程图

表 21-4　二级程序汇总表

序号	程序名称	程序号	功能	上级程序
1	初始化程序	CHUSH		MAIN
2	1＃码垛程序	PLT199		MAIN
3	2＃码垛程序	PLT299		MAIN
4	报警程序	BJ100		MAIN

其中,1＃码垛程序与2＃码垛程序内又各自可按层数分为10个子程序。部分子程序如表21-5所示。

表 21-5　三级程序汇总表

序号	程序名称	程序号	功能	上级程序
16	2＃1层码垛	PLT21		PLT299
17	2＃2层码垛	PLT22		PLT299
18	2＃3层码垛	PLT23		PLT299
19	2＃4层码垛	PLT24		PLT299
20	2＃5层码垛	PLT25		PLT299
21	2＃6层码垛	PLT26		PLT299
22	2＃7层码垛	PLT27		PLT299

续表

序号	程序名称	程序号	功能	上级程序
23	2#8 层码垛	PLT28		PLT299
24	2#9 层码垛	PLT29		PLT299
25	2#10 层码垛	PLT210		PLT299

2. 程序汇总表

经过程序结构分析,需要编制的程序如表 21-6 所示。

表 21-6　主程序子程序一览表

序　　号	程 序 名 称	程　序　号	功　　能	上 级 程 序
1	主程序	MAIN		
2	初始化程序	CHUSH		MAIN
3	1#码垛程序	PLT199		MAIN
4	2#码垛程序	PLT299		MAIN
5	报警程序	BJ100		MAIN
6	1#1 层码垛	PLT11		PLT199
7	1#2 层码垛	PLT12		PLT199
8	1#3 层码垛	PLT13		PLT199
9	1#4 层码垛	PLT14		PLT199
10	1#5 层码垛	PLT15		PLT199
11	1#6 层码垛	PLT16		PLT199
12	1#7 层码垛	PLT17		PLT199
13	1#8 层码垛	PLT18		PLT199
14	1#9 层码垛	PLT19		PLT199
15	1#10 层码垛	PLT110		PLT199
16	2#1 层码垛	PLT21		PLT299
17	2#2 层码垛	PLT22		PLT299
18	2#3 层码垛	PLT23		PLT299
19	2#4 层码垛	PLT24		PLT299
20	2#5 层码垛	PLT25		PLT299
21	2#6 层码垛	PLT26		PLT299
22	2#7 层码垛	PLT27		PLT299
23	2#8 层码垛	PLT28		PLT299
24	2#9 层码垛	PLT29		PLT299
25	2#10 层码垛	PLT210		PLT299

经过实验,可以将每一层的运动程序编制为一个子程序,在每一子程序中都重新定义 PLT(矩阵)规格,而且每一层的矩阵位置点也确实与上下一层各不相同。主程序就是顺序调用子程序,这样的编程简洁明了,同时也不受 PLT 指令数量的限制。

3. 主程序 MAIN

根据图 21-2 主程序流程图编制的主程序如下。

```
1  CallP"CHUSH"'——调用初始化程序。
```

```
 *LAB1'——程序分支标志。
If m_IN(12)=0  Then'——进行输送带有料/无料判断。
 GOTO LAB2'——如果输送带无料则跳转到*LAB2。
ELSE'——否则往下执行。
ENDIF'——判断语句结束。
If m_IN(14)=1 Then'——进行1#码垛位有料/无料(是否码垛完成)判断。
 GOTO LAB3'——如果1#码垛位有料(码垛完成)则跳转到*LAB3。
ELSE'——否则往下执行。
ENDIF'——判断语句结束。
CallP"PLT99"'——调用1#码垛程序。
 *LAB4'——程序结束标志。
END'——程序结束。
 *LAB2 输送带无料程序分支。
If m_IN(13)=1 Then'——进行待料时间判断。
m_OUT(13)=1'——如果待料时间超长则发出报警。
 GOTO  *LAB4'——结束程序。
ELSE'——否则重新检测输送带有料/无料。
GOTO  *LAB1'——跳转到*LAB1行。
ENDIF'——判断选择语句结束。
 *LAB3'—— 1#垛位有料程序分支,转入对2#码垛位的处理。
If m_IN(15)=1 Then'——如果2#垛位有料,则报警。
 m_OUT(13)=1'——指令输出信号13=ON。
GOTO  *LAB4'——结束程序。
ELSE'——否则
CallP"PLT199"'——调用2#码垛程序。
ENDIF'——判断选择语句结束。
END'——程序结束。
```

4. 1#垛码垛程序PLT99

1#垛码垛程序PLT99又分为10个子程序。每一层的码垛分为一个子程序。这是因为其一包装箱需要错层布置,防止垮塌;其二每一层的高度在增加,需要设置Z轴坐标。

```
CallP"PLT11"'——调用第1层码垛程序。
Dly 1'——暂停1s。
CallP"PLT12"'——调用第2层码垛程序。
Dly 1'——暂停1s。
CallP"PLT13"'——调用第3层码垛程序。
Dly 1'——暂停1s。
CallP"PLT14"'——调用第4层码垛程序。
Dly 1'——暂停1s。
CallP"PLT15"'——调用第5层码垛程序。
Dly 1'——暂停1s。
CallP"PLT16"'——调用第6层码垛程序。
Dly 1'——暂停1s。
CallP"PLT17"'——调用第7层码垛程序。
Dly 1'——暂停1s。
CallP"PLT18"'——调用第8层码垛程序。
Dly 1'——暂停1s。
CallP"PLT19"'——调用第9层码垛程序。
Dly 1'——暂停1s。
CallP"PLT110"'——调用第10层码垛程序。
```

```
20 End'——程序结束。
```

5. 码垛程序 PLT11(1#垛第1层)

码垛程序 PLT11 是1#垛第1层的码垛程序,在这个程序中,使用了专用的码垛指令,用于确定每一格的定位位置,是这个程序的关键点。

图 21-3 是1#1层码垛子程序的流程图。

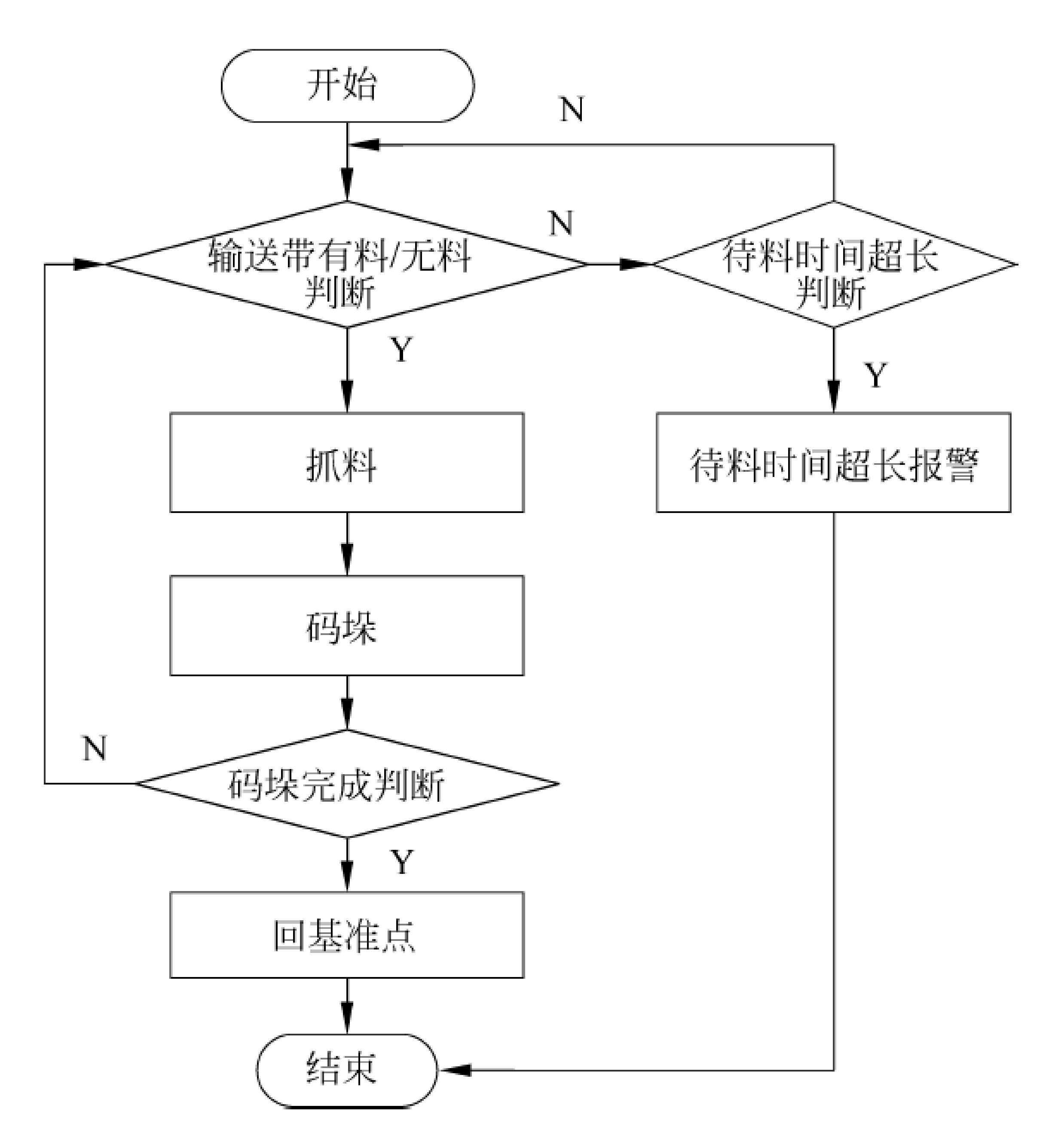


图 21-3　1#1层码垛子程序的流程图

在码垛程序 PLT11 中,其运动点位如图 21-4 所示。

48 终点B	47	46	45	44	43 对角点
37	38	39	40	41	42
36	35	34	33	32	31
25	26	27	28	29	30
24	23	22	21	20	19
13	14	15	16	17	18
12	11	10	9	8	7
1 起点	2	3	4	5	6 终点A

图 21-4　Plt 指令定义托盘位置图

```
1  Servo On'——伺服 ON。
2  Ovrd 20'——设置速度倍率。
```

```
3  '——以下对托盘 1 各位置点进行定义
4  P10 = P_01 + ( + 0.00, + 0.00, + 0.00, + 0.00, + 0.00, + 0.00) '——起点。
5  P11 = P10 + ( + 0.00, + 100.00, + 0.00, + 0.00, + 0.00, + 0.00) '——终点 A。
6  P12 = P10 + ( + 140.00, + 0.00, + 0.00, + 0.00, + 0.00, + 0.00) '——终点 B。
7  P13 = P10 + ( + 140.00, + 100.00, + 0.00, + 0.00, + 0.00, + 0.00) '——对角点,参见图 21 - 4。
8  DefPlt1, P10, P11,P12,P13,6,8,1 '——定义托盘 1。
9  m1 = 1 '——M1 表示各位置点。
10  * LOOP'循环程序起点标志
11 If m_IN(11) = 0 Then * LAB1 '——输送带有料/无料判断。如果无料,则跳转到 * LAB1 程序分支
处,否则往下执行。
12 Mov P1, - 50 '——移动到输送带位置点准备抓料。
13 Mvs P1 '——前进到 P1 点。
14 m_OUT(12) = 1 '——指令吸盘 = ON。
15 WAIT m_IN(12) = 1 '——等待吸盘 = ON。
16 Dly 0.5 '——暂停 0.5s。
17 Mvs, - 50 '——退回到 P1 点的"近点"。
18 P100 = Plt1,m1 '——以变量形式表示托盘 1 中的各位置点。
19 Mvs P100, - 50 '——运行到码垛位置点准备卸料。
20 Mvs P100 '——前进到 P100 点。
21 m_OUT(12) = 0 '——指令吸盘 = OFF,卸料。
22 WAIT m_IN(12) = 0 '——等待卸料完成。
23 Dly 0.3 '——暂停 0.3s。
24 Mvs, - 50 '——退回到 P100 点的"近点"。
25 m1 = m1 + 1 '——变量加 1。
26 If m1 <= 48  Then * LOOP'——判断:如果变量小于等于 48,则继续循环。
27 '——否则移动到输送带待料。
28 Mov P1, - 50 '——移动到输送带位置点准备抓料。
29 End'——程序结束。
30  * LAB1 '——程序分支标志
31 If m_IN(12) = 1 Then m_OUT(10) = 1 '——如果待料时间超长,则报警。
32 '——否则重新进入循环 * LOOP
33  GOTO * LOOP'——跳转到" * LOOP"行。
34 End'——程序结束。
```

6. 码垛程序 PLT12(1#垛第 2 层)

```
1  Servo On'——伺服 ON。
2  Ovrd 20 '——设置速度倍率。
3  '——以下对托盘 2 各位置点进行定义。注意,由于是错层布置,各起点、终点、对角点位置要重新
计算,而且抓手要旋转一个角度。
4  P10 = P_01 + ( + 0.00, + 0.00, + 10 * .00, + 0.00, + 0.00, + 90) '——起点
5  P11 = P10 + ( + 0.00, + 10 * .00, + 0.00, + 0.00, + 0.00, + 90) '——终点 A
6   P12 = P10 + ( + 14 * .00, + 0.00, + 0.00, + 0.00, + 0.00, + 90) '——终点 B
7  P13 = P10 + ( + 14 * .00, + 108.00, + 0.00, + 0.00, + 0.00, + 90) '——对角点
(省略)
```

码垛程序 PLT12(1#垛第 2 层)与码垛程序 PLT11(1#垛第 1 层)在结构形式上完全相同。唯一区别是托盘 2 的起点坐标在 Z 向上比第 1 层多一“层高”数据。注意程序中序号

第 4 行，其中，Z 向数值比码垛程序 PLT11 多一“层高”数值。由于是错层布置，各起点、终点、对角点位置要重新计算，而且抓手要旋转一个角度。

其余各层程序 PLT12～PLT110 均做如此处理。

21.4 结 束 语

机器人在码垛中的应用主要使用 PLT 指令，但实质上 PLT 指令只是一个定义矩阵格中心位置的指令。由于实际码垛一般需要错层布置，所以不能一个 PLT 指令用到底，每一层的位置都需要重新定义，然后使用循环指令反复执行抓取，而且必须作为一个完整的系统工程来考虑。

21.5 需要思考的问题

(1) 如何根据项目要求提出控制系统的配置方案？

(2) 如何分析主程序流程图构建主程序结构？

(3) 多层码垛每层工件的摆放位置是否相同？

(4) 码垛指令的实质动作是什么？

(5) 如何使用循环指令完成码垛动作？

第 22 章 第 22 日——机器人在手机检测生产线上的应用

【学习目的】

本章通过一个实际案例,学习机器人在流水线检测项目中的应用。学习案例重点是学习如何规划工艺流程,编制程序流程结构,如何调用子程序,如何使用判断-执行语句进行程序跳转。也要学习面对项目如何提出解决方案和构建控制系统。

22.1 项目综述

某手机检测生产线项目是机器人抓取手机(以下简称工件)进行检验,其工作过程如下:工件在流水线上,要求机器人抓取工件置于检验槽中,检验合格再抓取回流水线进入下一道工序。如果一次检验不合格,再抓取工件进入另外一个检验槽。共检验三次,如果全不合格则放置在废品槽中。设备布置如图 22-1 所示。

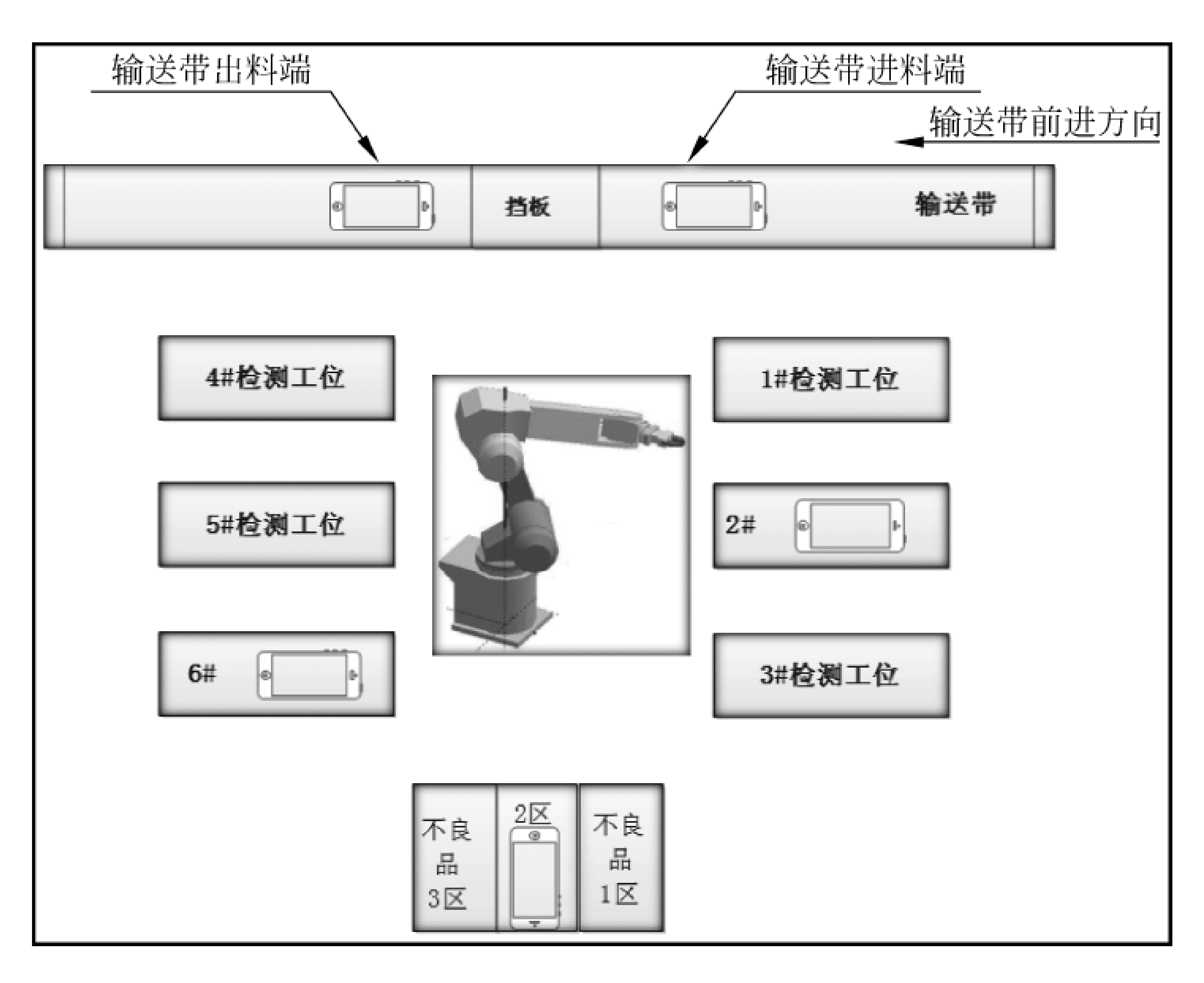


图 22-1 工程项目设备布置图

22.2 解决方案

经过技术经济可行性分析，决定采用如下方案。

（1）配置机器人一台作为工作中心，负责工件抓取搬运。机器人配置32点输入32点输出的I/O卡。选取三菱RV-2F机器人，该机器人搬运重量为2kg，最大动作半径504mm，可以满足工作要求。

（2）示教单元：R33TB（必须选配，用于示教位置点）。

（3）机器人选件：输入输出信号卡2D-TZ368（用于接收外部操作屏信号和控制外围设备动作）。

（4）选用三菱PLC FX3U-48MR作主控系统，用于控制机器人的动作并处理外部检测信号。

（5）配置AD模块FX3U-4AD用于接收检测信号。产品检测仪给出模拟信号，由AD模块处理后送入PLC做处理及判断。

（6）触摸屏选用GS2110。触摸屏可以直接与机器人相连接，直接设置和修改各工艺参数，发出操作信号。

22.2.1 硬件配置

硬件配置如表22-1所示。

表22-1 硬件配置一览表

序　　号	名　　称	型　　号	数　　量	备　　注
1	机器人	RV-2F	1	三菱
2	简易示教单元	R33TB	1	三菱
3	输入输出卡	2D-TZ368	1	三菱
4	PLC	FX3U-48MR	1	三菱
5	AD模块	FX3U-4AD	2	三菱
6	GOT	GS2110-WTBD	1	三菱

22.2.2 输入输出点分配

根据项目要求，需要配置的输入输出信号如表22-2和表22-3所示。在机器人一侧需要配置I/O卡。I/O卡型号为TZ-368。TZ-368的地址编号是机器人识别的I/O地址。

1. 输入信号地址分配

表22-2 输入信号地址一览表

序　　号	输入信号名称	输入地址（TZ-368）
1	自动启动	3
2	自动暂停	0
3	复位	2
4	伺服ON	4

续表

序　　号	输入信号名称	输入地址(TZ-368)
5	伺服 OFF	5
6	报警复位	6
7	操作权	7
8	回退避点	8
9	机械锁定	9
10	气压检测	10
11	输送带正常运行检测	11
12	输送带进料端有料无料检测	12
13	输送带出料端有料无料检测	13
14	1 工位有料无料检测	14
15	2 工位有料无料检测	15
16	3 工位有料无料检测	16
17	4 工位有料无料检测	17
18	5 工位有料无料检测	18
19	6 工位有料无料检测	19
20	1 工位检测合格信号	20
21	2 工位检测合格信号	21
22	3 工位检测合格信号	22
23	4 工位检测合格信号	23
24	5 工位检测合格信号	24
25	6 工位检测合格信号	25
26	1#废料区有料无料检测	26
27	2#废料区有料无料检测	27
28	3#废料区有料无料检测	28
29	抓手夹紧到位	29
30	抓手松开到位	30

2. 输出信号地址分配

表 22-3　输出信号地址一览表

序　　号	输出信号名称	输出信号地址(TZ-368)
1	机器人自动运行中	0
2	机器人自动暂停中	4
3	急停中	5
4	报警复位	2
5	抓手夹紧	11
6	抓手松开	12

3. 数值型变量 M 分配

由于本项目中机器人程序复杂，为编写程序方便，预先分配使用数值型变量和位置点的范围。数值型变量分配如表 22-4 所示。

表 22-4　数值型变量 M 分配一览表

序　　号	数值型变量名称	应 用 范 围
1	M1～M99	主程序
2	M100～M199	上料程序
3	M200～M299	卸料程序
4	M300～M499	不良品检测程序
5	M201～M206	1～6 工位有料无料检测
6	M221～M226	1～6 工位检测次数

4. 位置变量 P

位置变量 P 分配如表 22-5 所示。

表 22-5　位置变量 P 分配一览表

序　　号	位置变量名称	应 用 范 围	类　　型
1	P_30	机器人工作基准点	全局
2	P_10	输送带进料端位置	全局
3	P_20	输送带出料端位置	全局
4	P_01	1＃工位位置点	全局
5	P_02	2＃工位位置点	全局
6	P_03	3＃工位位置点	全局
7	P_04	4＃工位位置点	全局
8	P_05	5＃工位位置点	全局
9	P_06	6＃工位位置点	全局
10	P_07	1＃不良品区位置点	全局
11	P_08	2＃不良品区位置点	全局
12	P_09	3＃不良品区位置点	全局

22.3　编　　程

22.3.1　总流程

1. 总的工作流程

由于机器人程序复杂，应该首先编制流程图，根据流程图编制程序流程及程序框架。编制流程图时，需要考虑周全，确定最优工作路线，这样编程才能事半功倍。

总的工作流程如图 22-2 所示。

(1) 系统上电或启动后，首先进入“初始化”程序，包括检测输送带是否启动，启动气泵并检测气压及报警程序。

(2) 进入卸料工序，只有先将测试区的工件搬运回输送线上，才能够进行下一工步。

(3) 在卸料工步执行完毕后，进入“不良品处理工序”。在“不良品处理工序”中，要对检测不合格的产品执行三次检测，三次不合格才判定为不良品。从机器人动作来看，要将同一工件置于不同的三个测试工位中进行测试。测试不合格才将工件转入“不良品区”。执行

“不良品处理工步”也是要空出“测试区”。

(4) 经过“卸料工步”和“不良品处理工步”后，测试区各工位已经最大限度空出，所以执行“上料工步”。

(5) 如果工作过程中发生机器人系统的报警，机器人会停止工作。外部也配置有“急停按钮”。按下“急停按钮”后，系统立即停止工作。

(6) 总程序可以设置为“反复循环类型”，即启动之后反复循环执行，直到接收到停止指令。图 22-2 是总工作流程图。

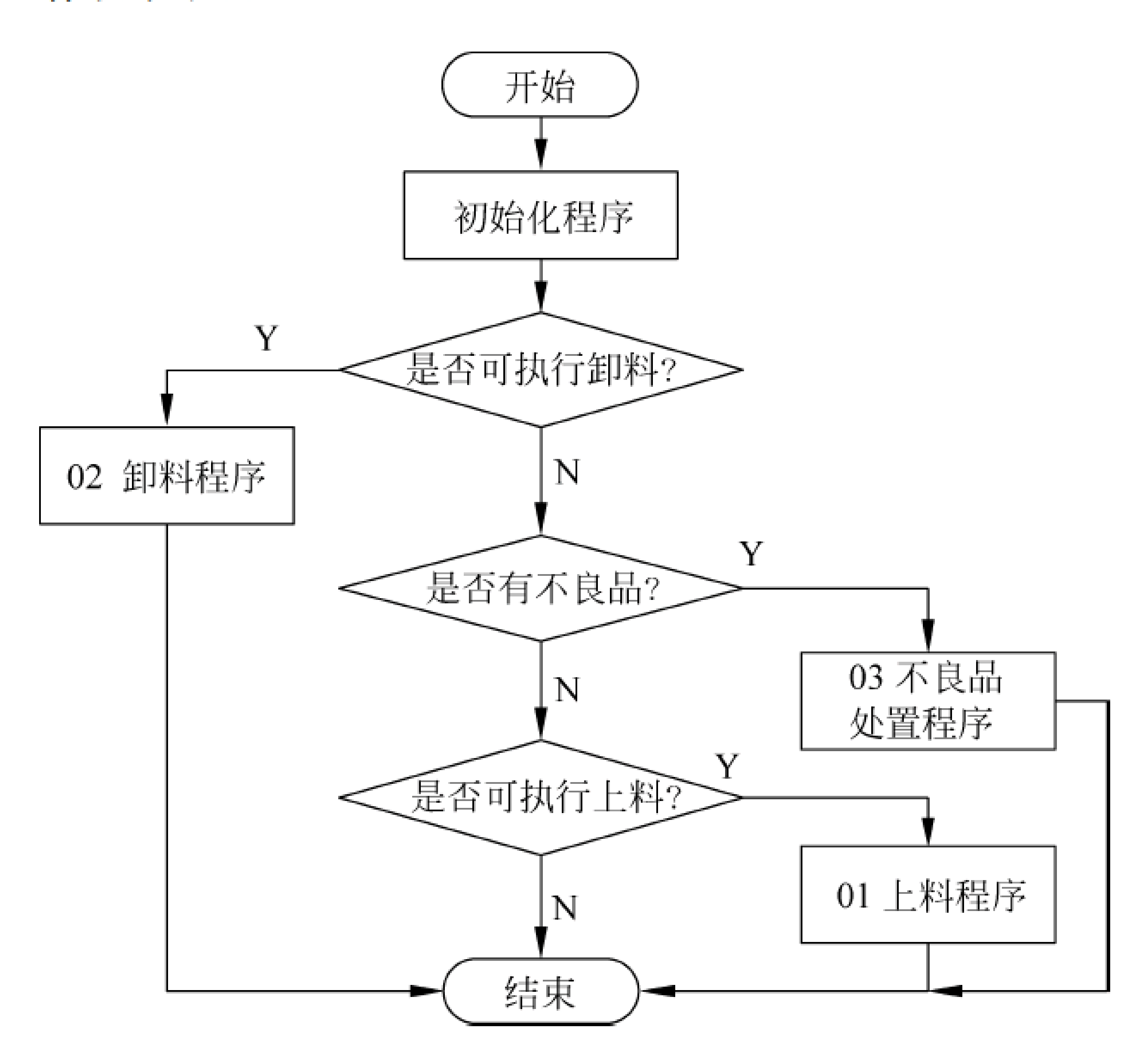


图 22-2　总工作流程图

2. 主程序 MAIN

根据总流程图编制的主程序 MAIN 如下。

```
CALLP "CSH"'——调用初始化程序。
'——进入卸料工步判断。
IF M210 = 6 THEN *LAB2'——如果全部工位检测不合格则跳*LAB2。
IF M_IN(13) = 1 THEN * LAB2'——如果输送带出口段有料则跳到* LAB2。
CALLP "XIEL"'——调用卸料程序。
GOTO *LAB4'——跳转到*LAB4 行。
'——*LAB2 "不良品工步"标记。
'——进入"不良品工步"工步判断。
IF M310 = 0 THEN *LAB3'——如果全部工位检测合格则跳*LAB3。
IF M310 = 6 THEN *LAB5'——如果全部工位检测不合格则跳*LAB5 报警程序。
CALLP "BULP"'——调用不良品处理程序。
GOTO *LAB4'——跳转到*LAB4 行。
*LAB3'——上料程序标记。
 IF M110 = 6 THEN *LAB4'——如果全部工位有料则跳到*LAB4。
IF M_IN(12) = 1 THEN *LAB4'——如果输送带进口段无料则跳到*LAB4。
CALLP "SL"'——调用上料程序。
*LAB4'——主程序结束标志。
```

```
18 END'——程序结束。
19 *LAB5'——报警程序。
20 CALLP"BAOJ"'——调用报警程序。
21 END'——程序结束。
```

22.3.2 初始化程序流程

初始化包括检测输送带是否启动，启动气泵并检测气压等工作。初始化的工作流程如图 22-3 所示。

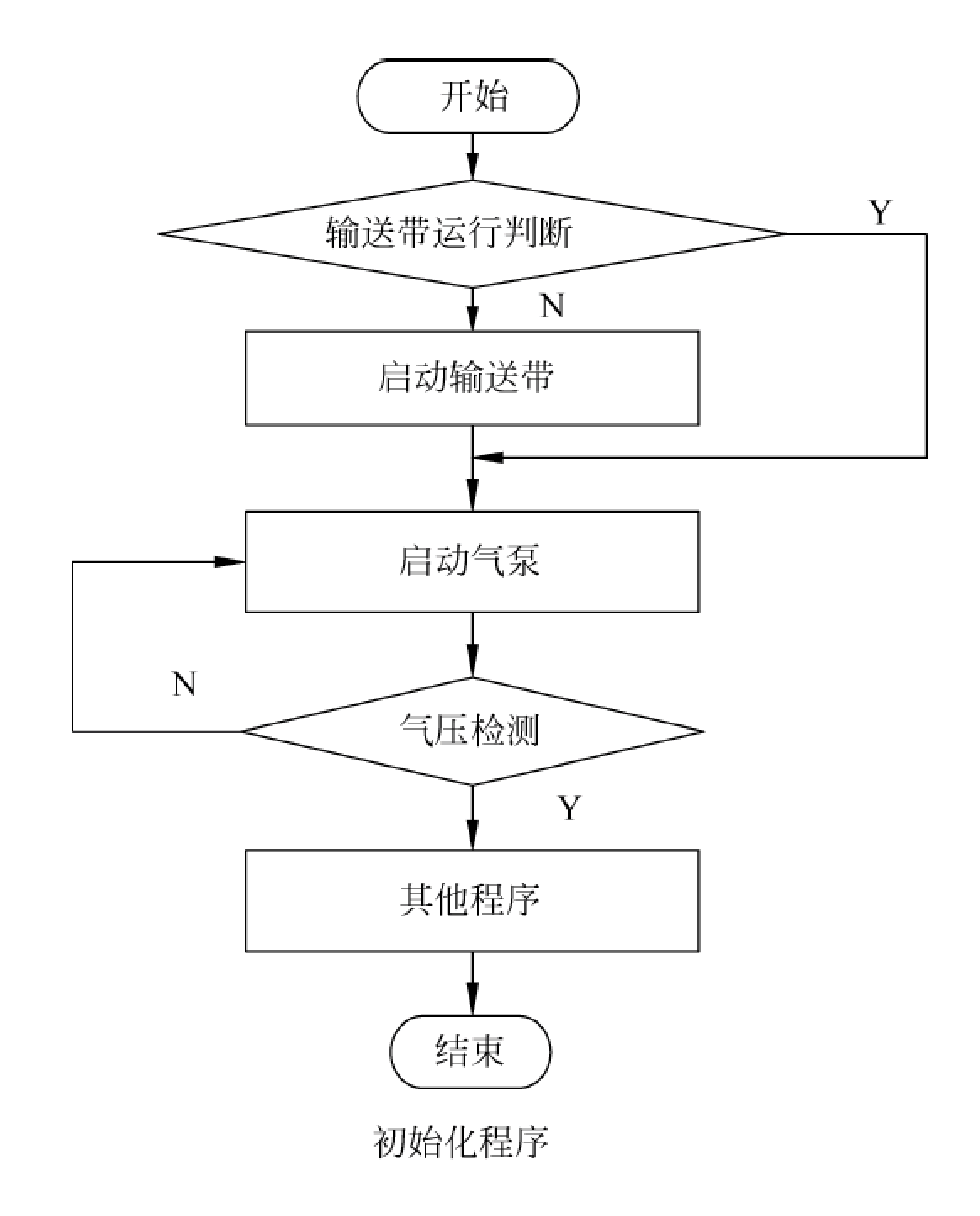


图 22-3　初始化工作流程

22.3.3 上料流程

1. 上料程序流程及要求

（1）上料程序必须首先判断：

① 输送带进口段上是否有料？

② 测试区是否有空余工位？

（2）如果不满足这两个条件，就结束上料程序返回主程序。

（3）如果满足这两个条件，则逐一判断空余工位，然后执行相应的搬运程序。

（4）由于上料动作必须将工件压入测试工位槽中，所以采用了机器人的柔性控制功能，在压入过程中如果遇到过大阻力，则机器人会自动做相应调整，这是关键技术之一。

（5）每一次搬运动作结束后，不是回到程序 END 处，而是回到程序起始处重新判断，直到 6 个工件全部装满。

2. 上料工步流程图

上料工步流程图如图 22-4 所示。

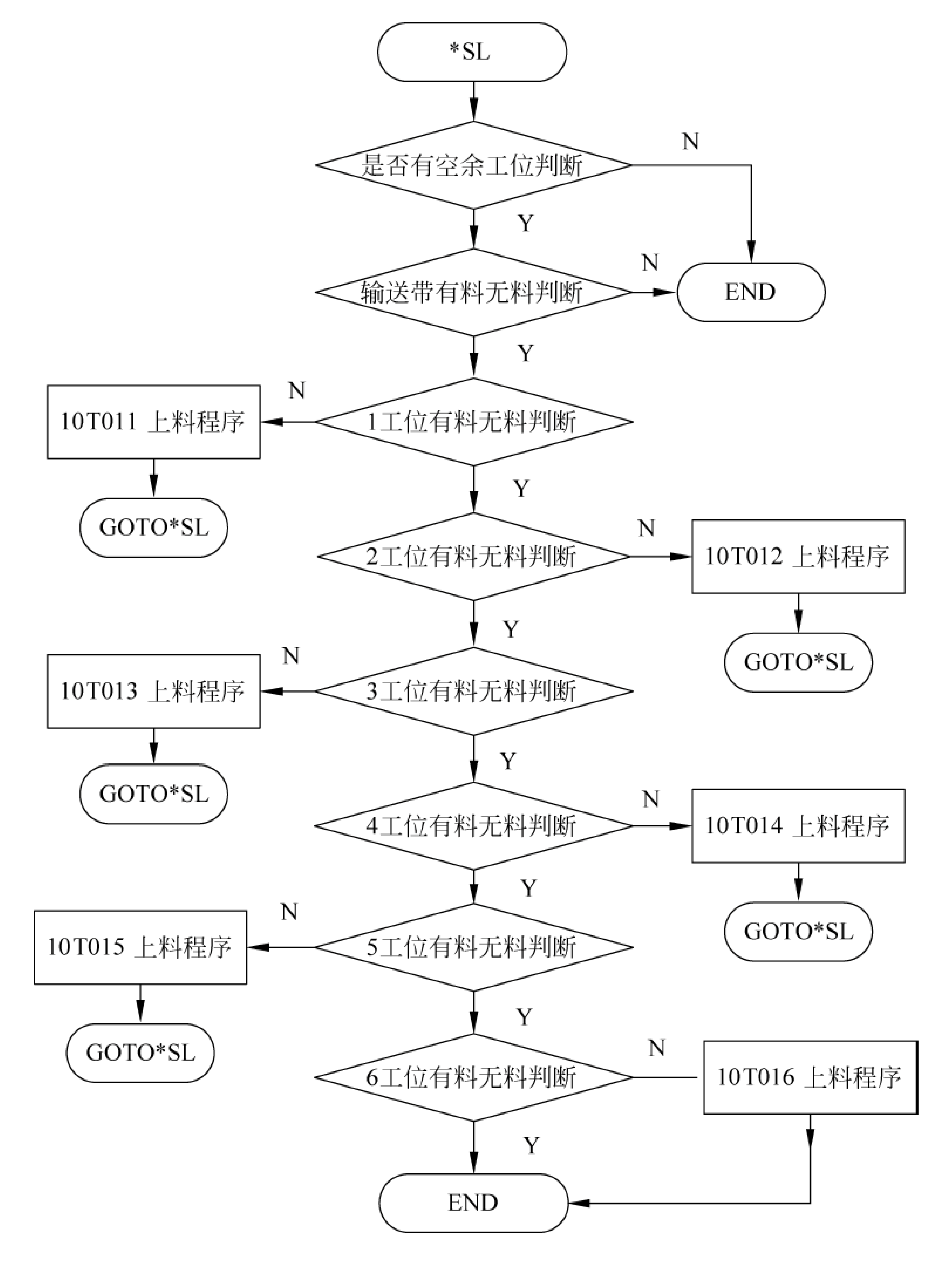


图 22-4　上料工步流程图

3. 上料程序 SL

```
 *SL 程序分支标签
M101 = M_In (14) '—— 1 工位有料无料检测信号。
M102 = M_In (15) '——2 工位有料无料检测信号。
M103 = M_In (16) '—— 3 工位有料无料检测信号。
M104 = M_In (17) '—— 4 工位有料无料检测信号。
M105 = M_In (18) '——5 工位有料无料检测信号。
M106 = M_In (19) '—— 6 工位有料无料检测信号。
M110 = M101 % + M102 % + M103 % + M104 % + M105 % + M106 % '——全部工位有料无料状态。
 IF M110 = 6 THEN * LAB'——如果全部工位有料则跳到程序结束。
 IF M_IN(12) = 1 THEN * LAB'——如果输送带无料则跳到程序结束。
```

```
'——如果 1 工位无料就执行上料程序"10TO11",否则进行 2 工位判断。
IF M_In (14)  = 0 THEN'——如果 1 工位无料
CALLP"10TO11"'——调用上料程序"10TO11"
 GOTO * SL'——跳转到 * SL
 ELSE'——否则
 ENDIF'——结束判断执行语句。
'如果 2 工位无料就执行上料程序"10TO12",否则进行 3 工位判断。
IF M_In (15)  = 0 THEN
CALLP"10TO12"
GOTO * SL
ELSE'(2)
ENDIF
'如果 3 工位无料就执行上料程序"10TO13",否则进行 4 工位判断。
IF M_In (16)  = 0 THEN
CALLP"10TO13"
ELSE'(3) 19 GOTO * SL
ENDIF
'如果 4 工位无料就执行上料程序"10TO14",否则进行 5 工位判断。
IF M_In (17)  = 0 THEN
CALLP"10TO14"
GOTO * SL
ELSE'(4)
ENDIF
'如果 5 工位无料就执行上料程序"10TO15",否则进行 6 工位判断。
IF M_In (18)  = 0 THEN
CALLP"10TO15"
GOTO * SL
ELSE'(5)
ENDIF
'如果 6 工位无料就执行上料程序"10TO16",否则结束上料程序。
IF M_In (19)  = 0 THEN
CALLP"10TO16"
ELSE'(6)
ENDIF'(6)
 * LAB
END
```

4. 程序 10TO11

本程序用于从输送带抓料到 1＃工作位,使用了柔性控制功能。

```
SERVOON'——伺服 ON。
OVRD100'——速度倍率 100％。
MOV P_10,50'——快进到输送带进料端位置点上方 50mm。
OVRD 20'——设置速度倍率为 20％。
MVS P_10'——慢速移动到输送带进料端位置点。
M_OUT(11) = 1'——抓手 ON。
WAIT M_IN(29) = 1'——等待抓手夹紧完成。
DLY 0.3'——暂停 0.3s。
MOV P_10,50。'——移动到输送带进料端位置点上方 50mm。
```

```
10 OVRD 100'——设置速度倍率为 100％。
11 MOV P_01, 50'——快进到 1＃工位位置点上方 50mm。
12 OVRD20'——设置速度倍率为 20％。
13 CmpG1, 1, 0.7, 1, 1, 1, ,'——设置各轴柔性控制增益值。
14 CmpPos, &B000100'——设置 Z 轴为柔性控制轴。
15 MVS P_01'——到 1＃工位位置点。
16 M_OUT(11)＝0'——松开抓手。
17 WAIT M_IN(30)＝1'——等待抓手松开完成。
18 DLY 0.3'——暂停 0.3s。
19 OVRD100'——设置速度倍率为 100％。
20 CmpOff'——关闭柔性控制功能。
21 MOV P_01, 50'——移动到 1＃工位位置点上方 50mm。
22 MOV P_30'——移动到基准点。
23  END'——主程序结束。
```

22.3.4 卸料工序流程

1. 卸料程序的流程及要求

（1）卸料程序必须首先判断：

① 输送带出口段上是否有料？

② 测试区是否有合格工件？

（2）如果不满足这两个条件，就结束卸料程序返回主程序。

（3）如果满足这两个条件，则逐一判断合格工件所在工位，然后执行相应的搬运程序。

（4）每一次搬运动作结束后，不是回到程序 END 处，而是回到程序起始处重新判断，直到全部合格工件被搬运到输送带上。

2. 卸料工步流程图

卸料工步流程图如图 22-5 所示。

3. 卸料程序 XIEL

```
1  *XIEL  程序分支标签
2  M201＝M_In (20)'—— 1 工位检测合格信号。
3  M202＝M_In (21)'——2 工位检测合格信号。
4  M203＝M_In (22)'—— 3 工位检测合格信号。
5  M204＝M_In (23)'——4 工位检测合格信号。
6  M205＝M_In (24)'—— 5 工位检测合格信号。
7  M206＝M_In (25)'—— 6 工位检测合格信号。
'——检测合格信号＝0 检测不合格信号＝1
8  M210＝M201＋M202＋M203＋M204＋M205＋M206
9  IF M210＝6 THEN *LAB20'——如果全部工位检测不合格则跳到程序结束。
3  IF M_IN(13)＝1 THEN * LAB20'——如果输送带有料则跳到程序结束。
4  '——如果 1 工位检测合格就执行卸料程序"21TO20",否则进行 2 工位判断。
5  IF M_In (20) ＝0 THEN
6  CALLP"21TO20"
7   GOTO *XIEL
8   ELSE'(1)
9   ENDIF
```

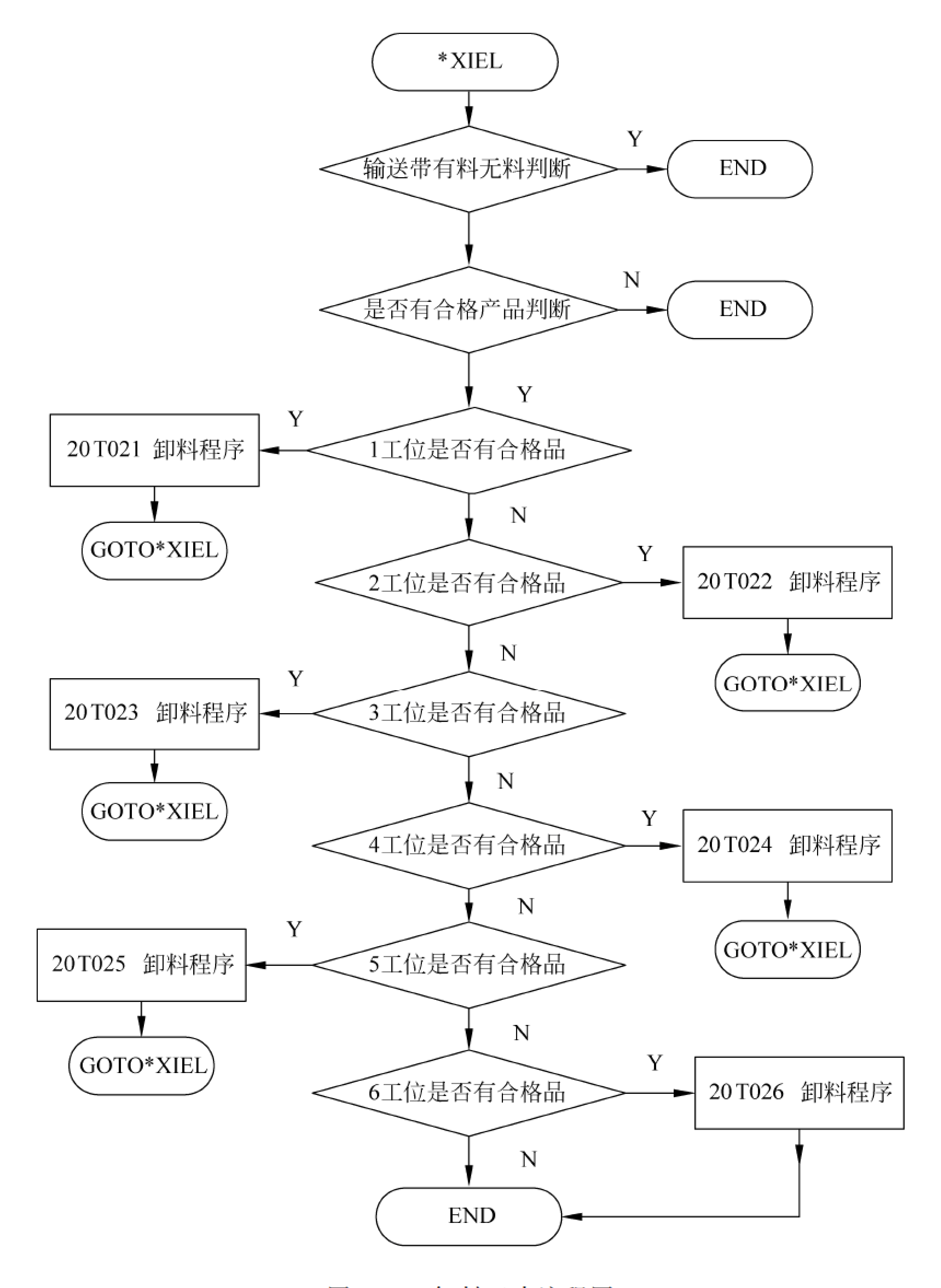


图 22-5 卸料工步流程图

```
10 '——'如果 2 工位检测合格就执行卸料程序"22T020",否则进行 3 工位判断。
11 IF M_In (21) = 0 THEN
12 CALLP"22T020"
13 GOTO * XIEL
14 ELSE'(2)
15 ENDIF
16 '——如果 3 工位检测合格就执行卸料程序"23T020",否则进行 4 工位判断。
17 IF M_In (22) = 0 THEN
18 CALLP"23T020"
19 GOTO * XIEL
20 ELSE'(3)
21 ENDIF
```

```
'——如果 4 工位检测合格就执行卸料程序"24TO20",否则进行 5 工位判断。
IF M_In (23)  = 0 THEN
CALLP"24TO20"
GOTO * XIEL
ELSE'(4)
ENDIF
'——如果 5 工位检测合格就执行卸料程序"25TO20",否则进行 6 工位判断。
IF M_In (24)  = 0 THEN
CALLP"25TO20"
GOTO * XIEL
ELSE'(5)
ENDIF
'——如果 6 工位检测合格就执行卸料程序"26TO20",否则 GOTO END。
IF M_In (25)  = 0 THEN
CALLP"25TO20"
ELSE'(6)
ENDIF'(6)
 * LAB20
END
```

22.3.5 不良品处理程序

1. 程序的要求及流程

（1）在“不良品处理工序”中，要对检测不合格的产品执行三次检测，三次不合格才判定为不良品。从机器人动作来看，要将同一工件置于不同的三个测试工位中进行测试。测试不合格才将工件转入“不良品区”。因此在“不良品处理工序”中：

① 首先判断有无不良品？无不良品则结束本程序返回上一级程序。

② 是否全部为不良品？如果全部为不良品，则必须报警，因为可能是出现了重大质量问题，需要停机检测。

（2）如果不满足以上条件，则逐一判断不良品所在工位，判断完成后，执行相应的搬运程序。

（3）在下一级子程序中，还需要判断是否有空余工位，并且标定检测次数，在检测次数为 3 时，将工件搬运到“不良品区”。

2. 不良品处理程序流程图

不良品处理程序流程图如图 22-6 所示。

3. 不良品处理程序 BULP

```
 * BULP 程序分支标签
M301 = M_In (20) '—— 1 工位检测合格信号。
M302 = M_In (21) '—— 2 工位检测合格信号。
M303 = M_In (22) '—— 3 工位检测合格信号。
M304 = M_In (23) '—— 4 工位检测合格信号。
M305 = M_In (24) '—— 5 工位检测合格信号。
M306 = M_In (25) '—— 6 工位检测合格信号。
'——检测合格信号 = 0,检测不合格信号 = 1。
```

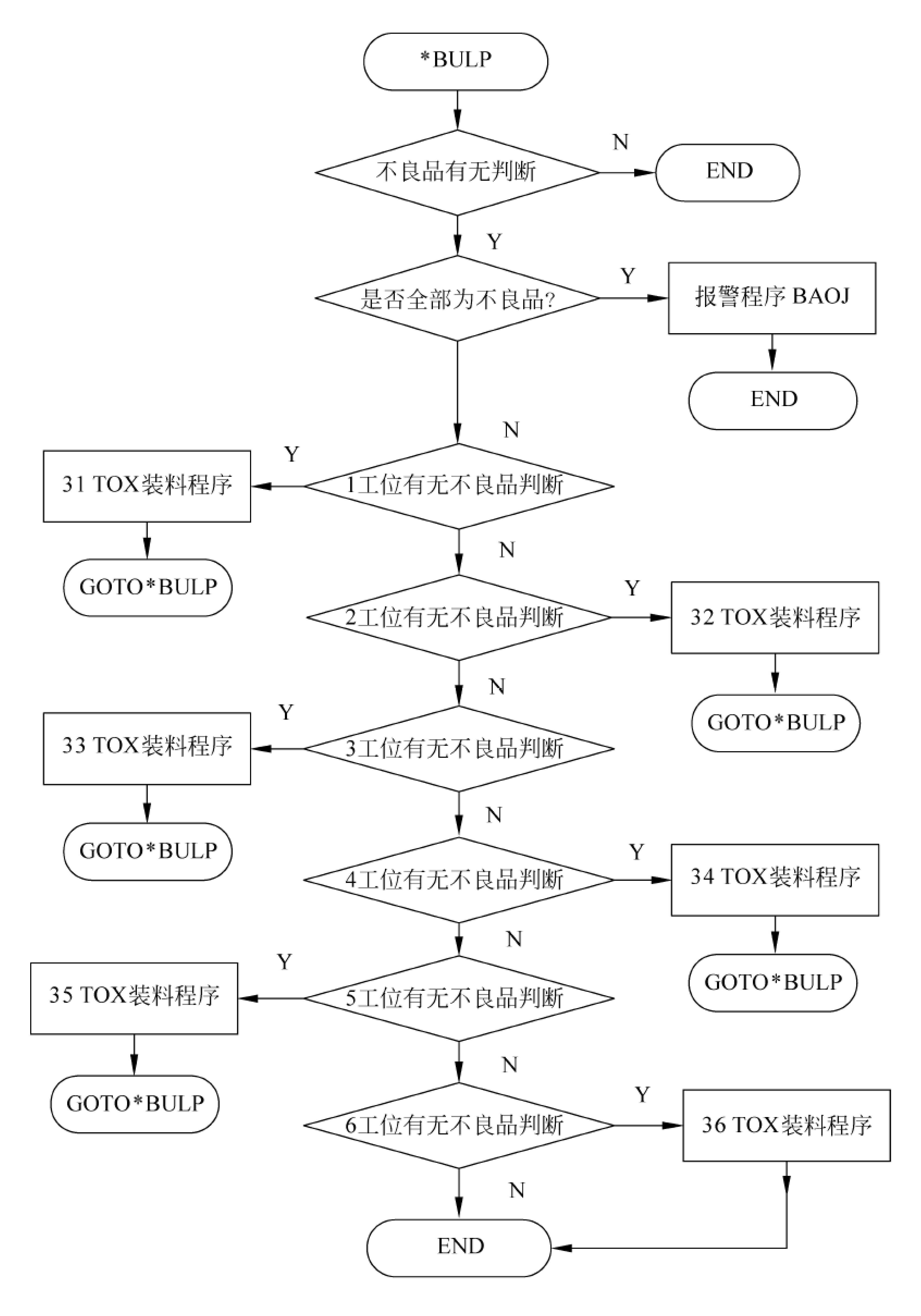


图 22-6　不良品处理流程图

```
M310 = M301 + M302 + M303 + M304 + M305 + M306
IF M310 = 0 THEN ＊LAB2'——如果全部工位检测合格则跳到＊LAB2(结束程序)。
IF M310 = 6 THEN ＊LAB3'——如果全部工位检测不合格则跳到报警程序。
'——如果 1 工位检测不合格就执行转运程序"31TOX",否则进行 2 工位判断。
IF M_In (20)  = 1 THEN
CALLP"31TOX"
 GOTO＊BULP ------ '回程序起始行。
 ELSE'(1)
 ENDIF
'——如果 2 工位检测不合格就执行转运程序"32TOX",否则进行 3 工位判断。
IF M_In (21)  = 1 THEN
```

```
12 CALLP"32TOX"
13 GOTO * BULP
14 ELSE'(2)
15 ENDIF
16 '——如果 3 工位检测不合格就执行转运程序"33TOX",否则进行 4 工位判断。
17 IF M_In (22) = 1 THEN
18 CALLP"33TOX"
19 GOTO * BULP
20 ELSE'(3)
21 ENDIF
22 '——如果 4 工位检测不合格就执行转运程序"34TOX",否则进行 5 工位判断。
23 IF M_In (23) = 1 THEN
24 CALLP"31TOX"
25 GOTO * BULP
26 ELSE'(4)
27 ENDIF
28 '——如果 5 工位检测不合格就执行转运程序"35TOX",否则进行 6 工位判断。
29 IF M_In (24) = 1 THEN
30 CALLP"35TOX"
31 GOTO * BULP
32 ELSE'(5)
33 ENDIF
34 '——如果 6 工位检测不合格就执行转运程序"36TOX",否则结束程序。
35 IF M_In (24) = 1 THEN
36 CALLP"36TOX"
37 ELSE'(6)
38 ENDIF'(6)
 * LAB2
END
39 * LAB3
40 END
```

22.3.6 不良品在1#工位的处理流程

1. 不良品在1#工位的处理程序(31TOX)流程

不良品在1#工位的处理程序(31TOX)流程图如图22-7所示。

(1) 当1#工位(包括2～5#工位)有不良品时,先要进行检测次数判断。工艺规定对每一工件要进行三次检测,如果三次检测都不合格,才可以判断为不良品。

(2) 当检测次数＝3时,进入"31TOFEIP"程序(将工件放入"不良品区")。

(3) 当检测次数＝2时,进入"31TO2X"子程序(将工件放入"其他工位"进行第三次检测)。

(4) 当检测次数＝0(第1次)时,进入"31TOX"子程序(将工件放入"其他工位"进行第二次检测)。

如果检测次数＝0(初始值),则顺序判断各工位有料无料状态,执行相应的搬运程序。

为此必须标定检测次数,从1#工位将工件转运到N#工位后必须对各工位的检验次

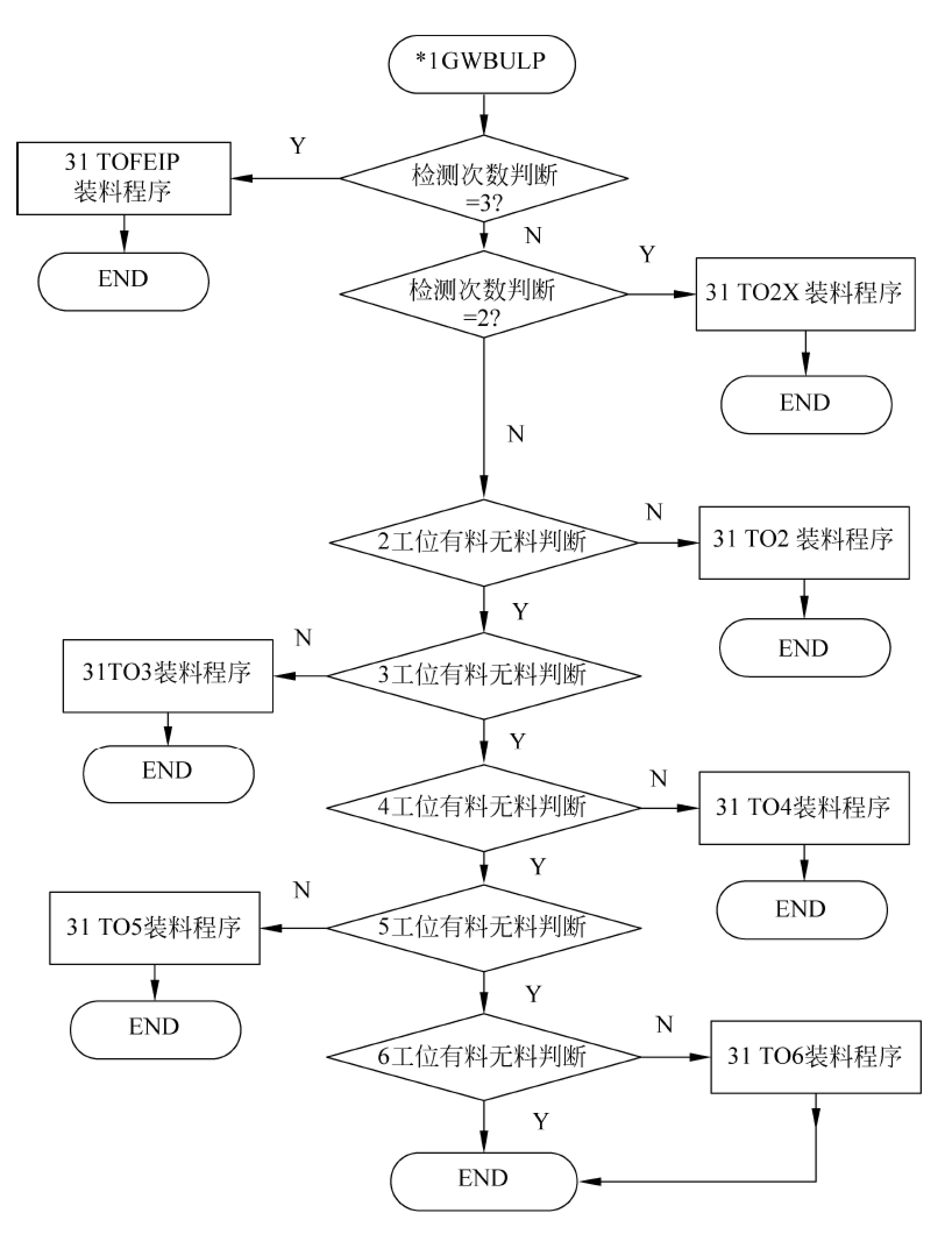


图 22-7　不良品在 1＃工位的处理程序(31TOX)流程

数进行标定,同时清掉 1＃工位的检测次数。

2. 不良品在 1＃工位的处理程序 31TOX

```
* 1GWEIBULP'——程序分支标签
'——如果检测次数 = 3 就执行不良品转运程序"31TOFEIP",否则进行下一判断。
IF M221 = 3 THEN CALLP "31TOFEIP"
'——如果检测次数 = 2 就执行转运程序"31TO2X",否则进行下一判断。
IF M221 = 2 THEN CALLP "31TO2X"
'——如果 2 工位无料就执行转运程序"31TO2",否则进行 3 工位判断。
IF M_In (14)  = 0 THEN
CALLP"31TO2"
M222 = 2'——标定 2 工位检测次数 = 2
M221 = 0'——标定 1 工位检测次数 = 0
GOTO * LAB2
ELSE'(1)
```

```
13 ENDIF
'——如果 3 工位无料就执行转运程序"31TO3",否则进行 4 工位判断。
14 IF M_In (15)  = 0 THEN
15 CALLP"31TO3"
16 M223 = 2'——标定 3 工位检测次数 = 2。
17 M221 = 0'——标定 1 工位检测次数 = 0。
18  GOTO * LAB2
19 ELSE'(2)
20 ENDIF
21 '——如果 4 工位无料就执行转运程序"31TO4",否则进行 5 工位判断。
22  IF M_In (17)  = 0 THEN
23 CALLP"31TO4"
24  M224 = 2'——标定 4 工位检测次数 = 2。
25 M221 = 0'——标定 1 工位检测次数 = 0。
26  GOTO * LAB2
27 ELSE'(3)
28 ENDIF
29 '——如果 5 工位无料就执行转运程序"31TO5",否则进行 6 工位判断。
30 IF M_In (18)  = 0 THEN
31 CALLP"31TO5"
32 M225 = 2'——标定 5 工位检测次数 = 2。
33  M221 = 0'——标定 1 工位检测次数 = 0。
34  GOTO * LAB2
35  ELSE'(4)
36  ENDIF
37 '——如果 6 工位无料就执行转运程序"31TO6",否则 GOTO END。
38  IF M_In (1)  = 0 THEN
39 CALLP"31TO6"
40 M226 = 2'——标定 5 工位检测次数 = 2。
41 M221 = 0'——标定 1 工位检测次数 = 0。
42 GOTO * LAB2
43  ELSE'(5)
44  ENDIF
45  '——如果 6 工位检测不合格就执行转运程序"36TOX",否则结束程序。
46  IF M_In (24)  = 1 THEN
47  CALLP"36TOX"
48  ELSE'(6)
49  ENDIF'(6)
50  * LAB2
51 END
52  * LAB3
53  END
```

3. 不良品在 1＃工位的向废品区的转运程序 31TOX

不良品在 2＃～6＃工位向废品区的转运程序与此类似。

(1) 流程图如图 22-8 所示。

(2) 程序可参见“31TOX”。

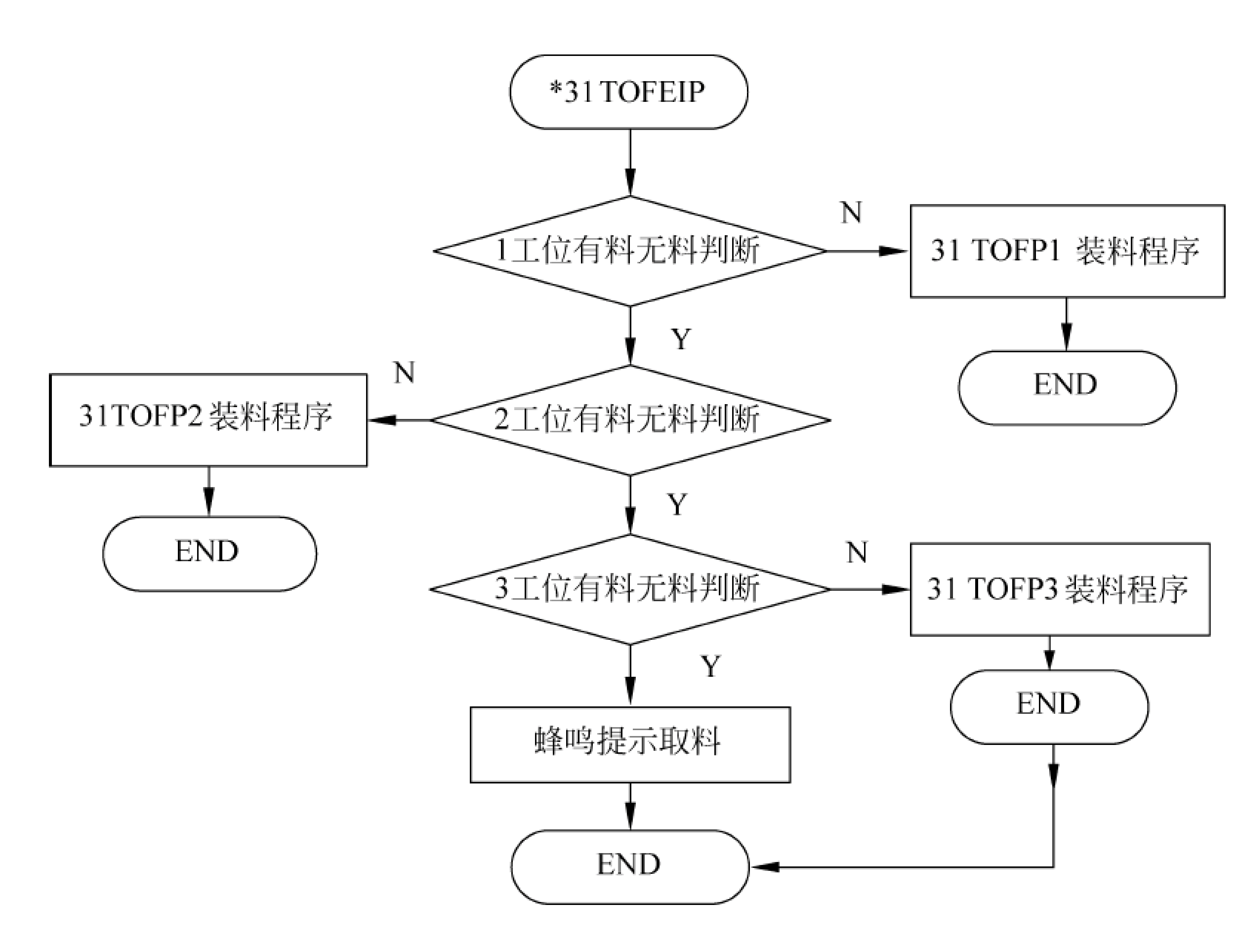


图 22-8　不良品在 1# 工位时向废品区的转运流程图

22.3.7　主程序子程序汇总表

由以上的分析可知，即使看似简单的搬运测试程序，也可以分解成许多子程序。对应于复杂的程序流程，将一段固定的动作编制为子程序是一种简单实用的编程方法。表 22-6 是程序汇总表。

表 22-6　主程序子程序汇总表

序　　号	程序名称	程　序　号	功　　能	上级程序
1	主程序	MAIN		
第 1 级子程序				
2	上料子程序	SL		MAIN
3	卸料子程序	XL		MAIN
4	不良品处理程序	BULP		MAIN
	报警程序	BAOJ		MAIN
第 2 级子程序　上料子程序所属子程序				
5	输送带到 1 工位	10TO11		SL
6	输送带到 2 工位	10TO12		SL
7	输送带到 3 工位	10TO13		SL
8	输送带到 4 工位	10TO14		SL
9	输送带到 5 工位	10TO15		SL
10	输送带到 6 工位	10TO16		SL
第 2 级子程序　卸料子程序所属子程序				
11	1 工位到输送带	21TO20		XL
12	2 工位到输送带	22TO20		XL

续表

序　　号	程序名称	程　序　号	功　　能	上级程序
13	3 工位到输送带	23TO20		XL
14	4 工位到输送带	24TO20		XL
15	5 工位到输送带	25TO20		XL
16	6 工位到输送带	26TO20		XL
第 2 级子程序　不良品处理程序所属子程序				
17	不良品从 1 工位转其他工位	31TOX		BULP
18	不良品从 2 工位转其他工位	32TOX		BULP
19	不良品从 3 工位转其他工位	33TOX		BULP
20	不良品从 4 工位转其他工位	34TOX		BULP
21	不良品从 5 工位转其他工位	35TOX		BULP
22	不良品从 6 工位转其他工位	36TOX		BULP
23	不良品从 1 工位转废品区程序	31TOFP		BULP
24	不良品从 2 工位转废品区程序	32TOFP		BULP
25	不良品从 3 工位转废品区程序	33TOFP		BULP
26	不良品从 4 工位转废品区程序	34TOFP		BULP
27	不良品从 5 工位转废品区程序	35TOFP		BULP
28	不良品从 6 工位转废品区程序	36TOFP		BULP
第 3 级子程序				
29	不良品从 1 工位转 2 工位程序	31TO32		31TOX
30	不良品从 1 工位转 3 工位程序	31TO33		31TOX
31	不良品从 1 工位转 4 工位程序	31TO34		31TOX
32	不良品从 1 工位转 5 工位程序	31TO35		31TOX
33	不良品从 1 工位转 6 工位程序	31TO36		31TOX
34	不良品从 2 工位转 1 工位程序	32TO31		32TOX

续表

序　　号	程序名称	程　序　号	功　　能	上级程序
35	不良品从 2 工位转 3 工位程序	32TO33		32TOX
36	不良品从 2 工位转 4 工位程序	32TO34		32TOX
37	不良品从 2 工位转 5 工位程序	32TO35		32TOX
38	不良品从 2 工位转 6 工位程序	32TO36		32TOX
39	不良品从 3 工位转 1 工位程序	33TO31		33TOX
40	不良品从 3 工位转 2 工位程序	33TO32		33TOX
41	不良品从 3 工位转 4 工位程序	33TO34		33TOX
42	不良品从 3 工位转 5 工位程序	33TO35		33TOX
43	不良品从 3 工位转 6 工位程序	33TO36		33TOX
44	不良品从 4 工位转 1 工位程序	34TO31		34TOX
45	不良品从 4 工位转 2 工位程序	34TO32		34TOX
46	不良品从 4 工位转 3 工位程序	34TO33		34TOX
47	不良品从 4 工位转 5 工位程序	34TO35		34TOX
48	不良品从 4 工位转 6 工位程序	34TO36		34TOX
49	不良品从 5 工位转 1 工位程序	35TO31		35TOX
50	不良品从 5 工位转 2 工位程序	35TO32		35TOX
51	不良品从 5 工位转 3 工位程序	35TO33		35TOX
52	不良品从 5 工位转 4 工位程序	35TO34		35TOX
53	不良品从 5 工位转 6 工位程序	35TO36		35TOX
54	不良品从 6 工位转 1 工位程序	36TO31		36TOX
55	不良品从 6 工位转 2 工位程序	36TO32		36TOX

续表

序　　号	程序名称	程　序　号	功　　能	上级程序
56	不良品从6工位转3工位程序	36TO33		36TOX
57	不良品从6工位转4工位程序	36TO34		36TOX
58	不良品从6工位转5工位程序	36TO35		36TOX
59	不良品从1工位转废品区1	31TOFP1		31TOFP
60	不良品从1工位转废品区2	31TOFP2		31TOFP
61	不良品从1工位转废品区3	31TOFP3		31TOFP
62	不良品从2工位转废品区1	32TOFP1		32TOFP
63	不良品从2工位转废品区2	32TOFP2		32TOFP
64	不良品从2工位转废品区3	33TOFP3		32TOFP
65	不良品从3工位转废品区1	33TOFP1		33TOFP
66	不良品从3工位转废品区2	33TOFP2		33TOFP
67	不良品从3工位转废品区3	33TOFP3		33TOFP
68	不良品从4工位转废品区1	34TOFP1		34TOFP
69	不良品从4工位转废品区2	34TOFP2		34TOFP
70	不良品从4工位转废品区3	34TOFP3		34TOFP
71	不良品从5工位转废品区1	35TOFP1		35TOFP
72	不良品从5工位转废品区2	35TOFP3		35TOFP
73	不良品从5工位转废品区3	36TOFP3		35TOFP
74	不良品从6工位转废品区1	36TOFP1		36TOFP
75	不良品从6工位转废品区2	36TOFP3		36TOFP
76	不良品从6工位转废品区3	36TOFP3		36TOFP

22.4 结 束 语

(1) 在工件检测项目中,编程的主要问题不是编制搬运程序,而是建立一个优化的程序流程。因此在进行程序编程初期,要与设备制造商的设计人员反复商讨工艺流程,在确认一个优化的工作流程后,再着手编制流程图和后续程序。

(2) 对每一段固定的动作必须将其编制为子程序,以简化编程工作和有利于对主程序的分析。

(3) 柔性控制技术在将工件压入检测槽时是关键技术。如果工件没有紧密放置在检测槽内,会影响检测结果。

22.5 需要思考的问题

(1) 在主程序中有几个二级子程序? 几个三级子程序?

(2) 本案例的处理流程是最佳流程吗? 能否提出别的处理流程?

(3) 为什么需要初始化流程?

(4) 什么是柔性控制技术?

(5) 主程序是循环执行程序吗?

第 23 章 第 23 日——编程指令的学习(5)

【学习目的】

在经过两个实际案例学习后,读者会感到前面学习的编程指令不够用,所以本章进行编程指令的深度学习。

23.1 Spd(Speed)——速度设置指令

1. 功能

本指令设置直线插补、圆弧插补时的速度,也可以设置最佳速度控制模式,以 mm/s 为单位设置。

2. 指令格式

```
Spd  <速度>
Spd  M_NSpd(最佳速度控制模式)
```

<速度>:单位为 mm/s。

3. 指令例句

```
1  Spd 100'——设置速度 = 100mm/s
2  Mvs P1'——前进到 P1 点
3  Spd M_NSpd'——设置初始值(最佳速度控制模式)
4  Mov P2'——前进到 P2 点
5  Mov P3'——前进到 P3 点
6  Ovrd 80'——设置速度倍率 = 80 %
7  Mov P4'——前进到 P4 点
Ovrd 100'——设置速度倍率 = 100 %
```

4. 说明

(1) 实际速度=操作面板倍率×程序速度倍率×Spd。

(2) M_NSpd 为初始速度设定值(通常为 10 000)。

23.2 JOvrd ——设置关节轴旋转速度的倍率

1. 功能

本指令用于设置以关节轴方式运行时的速度倍率。

2. 指令格式

```
JOvrd  <速度倍率>
```

3. 指令例句

```
1  JOvrd 50'——设置关节轴运行速度倍率 = 50 % 。
2  Mov P1'——前进到 P1 点。
3  J Ovrd M_NJOvrd'——设置关节轴运行速度倍率为初始值。
```

23.3 Accel——设置加减速阶段的“加减速度的倍率”

1. 功能

设置加减速阶段的“加减速度的倍率”(注意不是速度倍率)。

2. 指令格式

```
Accel <加速度倍率>,<减速度倍率>,<圆弧上升加减速度倍率>,<圆弧下降加减速度倍率>
```

3. 指令格式说明

(1) <加减速度倍率>——用于设置加减速度的“倍率”。

(2) <圆弧上升加减速度倍率>——对于 Mva 指令,用于设置圆弧段加减速度的“倍率”。

4. 指令例句

```
1  Accel 50,100'——假设标准加速时间为 0.2s,则加速度阶段倍率 = 50 % ,即 0.4s。减速度阶段倍
率 = 100 % ,即 0.2s。
2  Mov P1'——前进到 P1 点。
3  Accel 100,100'——假设标准加速时间为 0.2s,则加速度阶段倍率 = 100 % ,即 0.2s。减速度阶段
倍率 = 100 % ,即 0.2s。
4  Mov P2'——前进到 P2 点。
5  Def Arch 1, 10,10,25,25,1,0,0'——定义圆弧。
6  Accel 100,100,20,20,20,20'——设置圆弧上升下降阶段加减速度倍率。
7  Mva P3,1'——前进到 P3 点。
```

23.4 Cmp Jnt(Comp Joint)——指定关节轴进入“柔性控制状态”

1. 功能

本指令用于指定关节轴进入柔性控制状态。

2. 指令格式

```
Cmp  Jnt  <轴号>
```

<轴号>——轴号用一组二进制编码指定。&B000000 对应 654321 轴。

3. 指令例句

```
1  Mov P1'——前进到 P1 点。
2  CmpG 0.0,0.0,1.0,1.0, , , ,'——指定柔性控制度。
```

```
Cmp Jnt,&B11'——指定 J1 轴、J2 轴进入柔性控制状态。
Mov P2'——前进到 P2 点。
HOpen 1'——抓手动作。
Mov P1'——前进到 P2 点。
Cmp Off '——返回常规状态。
```

23.5 Cmp Pos——指定伺服轴进入"柔性控制工作模式"

1. 功能

本指令以直角坐标系为基准,指定伺服轴(CBAZYX)进入柔性控制工作模式。

2. 指令格式

```
Cmp  Pos,<轴号>
```

<轴号>:用一组二进制编码指定。&B000000 对应 CBAZYX 轴。

3. 指令例句

```
Mov P1'——前进到 P1 点。
CmpG 0.5, 0.5, 1.0, 0.5, 0.5, , ,'——指定柔性控制度。
Cmp Pos, &B011011'——设置 X,Y,A,B 轴进入柔性控制模式。
Mvs P2'——前进到 P1 点。
M_Out(10) = 1'——指令输出端子 10 = ON。
Dly 1.0'——暂停 1s。
HOpen 1'——抓手动作。
Mvs, -100'——后退 100。
Cmp Off'——返回常规状态。
```

23.6 Cmp Tool——指定伺服轴进入"柔性控制工作模式"

1. 功能

以 TOOL 坐标系为基准,指定伺服轴(CBAZYX)进入柔性控制工作模式。

2. 指令格式

```
Cmp  Tool,<轴号>
```

<轴号>:轴号用一组二进制编码指定。&B000000 对应 CBAZYX 轴。

3. 指令例句

```
Mov P1'——前进到 P1 点。
CmpG 0.5, 0.5, 1.0, 0.5, 0.5, , ,'——指定柔性控制度。
Cmp Tool, &B011011'——指定 TOOL 坐标系中的 X,Y,A,B 轴进入柔性控制工作模式。
Mvs P2'——前进到 P2 点。
M_Out(10) = 1'——指令输出端子 10 = ON。
Dly 1.0'——暂停 1s。
```

```
7  HOpen 1'——抓手动作。
8  Mvs, -100'——后退 100。
9  Cmp Off'——返回常规状态。
```

23.7 Cmp Off——解除机器人柔性控制工作模式

1. 功能

本指令用于解除机器人柔性控制工作模式。

2. 指令格式

```
Cmp  Off
```

3. 指令例句

```
1  Mov P1'——前进到 P1 点。
2  CmpG 0.5, 0.5, 1.0, 0.5, 0.5, , ,'——指定柔性控制度。
3  Cmp Tool, &B011011 ----- '指定 TOOL 坐标系中的 X,Y,A,B 轴进入柔性控制工作模式。
4  Mvs P2'——前进到 P2 点。
5  M_Out(10) = 1'——指令输出端子 10 = ON。
6  Dly 1.0'——暂停 1s。
7  HOpen 1'——抓手动作。
8  Mvs, -100'——后退 100。
9  Cmp Off'——机器人柔性控制工作模式 = OFF。
```

23.8 CmpG（Composition Gain）——设置柔性控制时各轴的增益

1. 功能

本指令用于设置柔性控制时各轴的柔性控制增益。

2. 指令格式

1）直角坐标系

```
CmpG  [<X 轴增益>]  [<Y 轴增益>]  [<Z 轴增益>]  [<A 轴增益>]  [<B 轴增益>]  [<C 轴增益>]
```

2）关节型

```
CmpG  [<J1 轴增益>]  [<J2 轴增益>]  [<J3 轴增益>]  [<J4 轴增益>]  [<J5 轴增益>]  [<J6 轴增益>]
```

3）说明

[<** 轴增益>]：用于设置各轴的柔性控制增益。常规状态＝1。以柔性控制增益＝1为基准进行设置。

3. 指令例句

```
CmpG , ,0.5, , , , ,'——设置 Z 轴的柔性控制增益 = 0.5,省略设置的轴用"逗号"分隔。
```

4. 说明

(1) 以指令位置与实际位置为比例,像弹簧一样产生作用力(实际位置越接近指令位置,作用力越小)。CmpG 就相当于弹性常数。

(2) 指令位置与实际位置之差可以由状态变量 M_CmpDst 读出,可用变量 M_CmpDst 判断动作(例如 PIN 插入)是否完成。

(3) 柔性控制增益调低时,动作位置精度会降低,因此必须逐步调整确认。

(4) 各型号机器人可以设置的最低"柔性度(增益)"如表 23-1 所示。

表 23-1 各型号机器人可以设置的最低"柔性度(增益)"

机　　型	cmp pos cmp tool 指令	cmp jnt 指令
RH-F 系列	0.20,0.20,0.20,0.20,0.20,0.20	0.01,0.01,0.20,0.01,1.00,1.00
RV-F 系列	0.01,0.01,0.01,0.01,0.01,0.01	

23.9 Oadl(Optimal Acceleration)——对应抓手及工件条件,选择最佳加减速模式的指令

1. 功能

本指令根据对应抓手及工件条件,选择最佳加减速时间,所以也称为最佳加减速模式选择指令。

2. 指令格式

```
Oadl  <On/Off>
```

Oadl On:最佳加减速模式=ON。

Oadl OFF:最佳加减速模式=OFF。

3. 指令例句

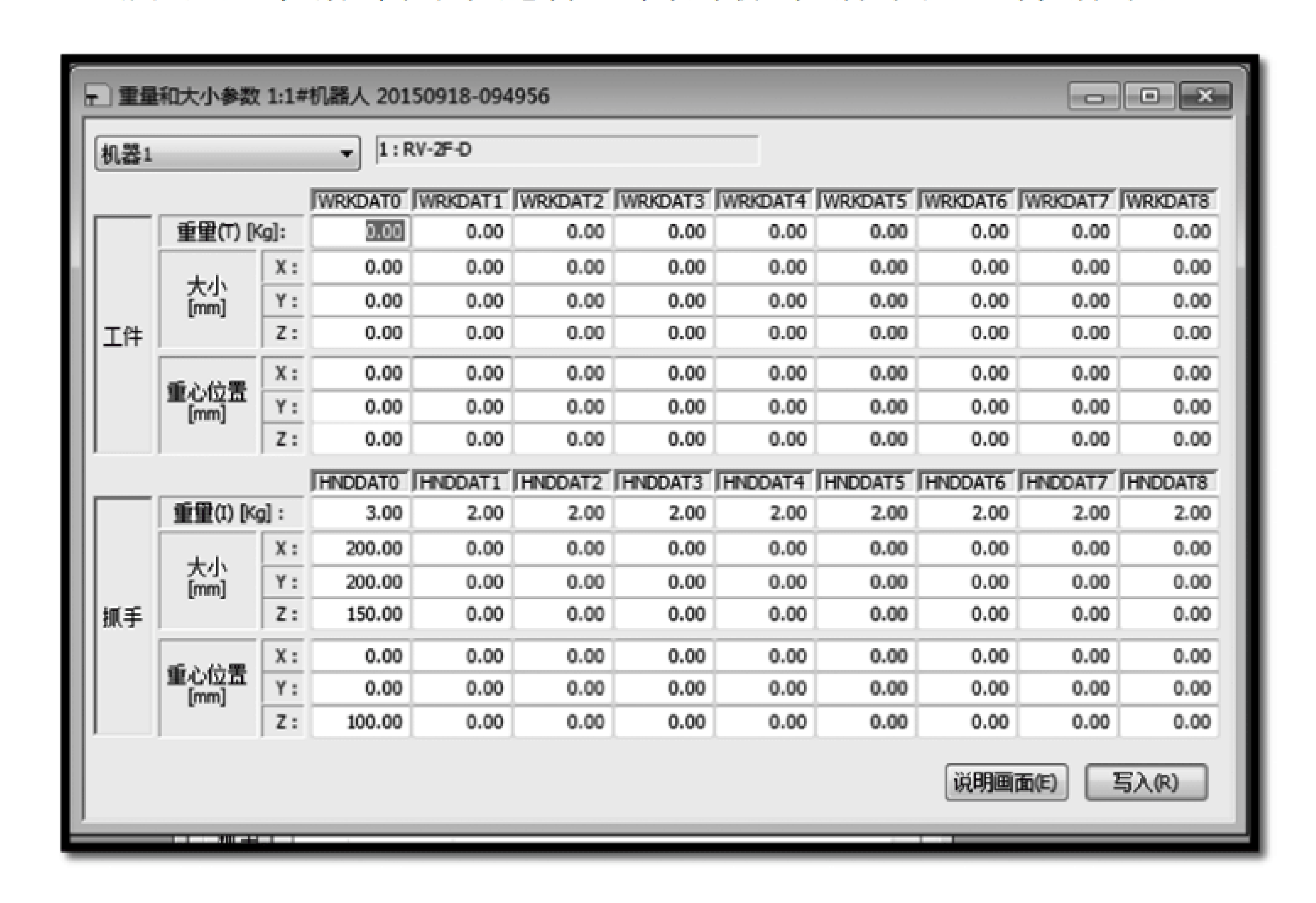
```
1  Oadl On '——最佳加减速模式 = ON。
2  Mov P1'——前进到 P1 点。
3  LoadSet 1'——设置抓手及工件类型。
4  Mov P2'——前进到 P2 点。
5  HOpen 1'——抓手动作。
6  Mov P3'——前进到 P3 点。
7  HClose 1'——抓手动作。
8  Mov P4'——前进到 P4 点。
9  Oadl Off '——最佳加减速模式 = OFF。
```

23.10 LoadSet(Load Set)——设置抓手、工件的工作条件

1. 功能

在实用的机器人系统配置完毕后,抓手及工件的重量、大小和重心位置通过参数已经设

置完毕，如图 23-1 所示。本指令用于选择不同的抓手编号及工件编号。



工件			WRKDAT0	WRKDAT1	WRKDAT2	WRKDAT3	WRKDAT4	WRKDAT5	WRKDAT6	WRKDAT7	WRKDAT8
工件	重量(T) [Kg]:		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	大小 [mm]	X :	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		Y :	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		Z :	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	重心位置 [mm]	X :	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		Y :	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		Z :	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

抓手			HNDDAT0	HNDDAT1	HNDDAT2	HNDDAT3	HNDDAT4	HNDDAT5	HNDDAT6	HNDDAT7	HNDDAT8
抓手	重量(I) [Kg] :		3.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00
	大小 [mm]	X :	200.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		Y :	200.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		Z :	150.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	重心位置 [mm]	X :	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		Y :	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		Z :	100.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

图 23-1　使用参数对抓手及工件重量及重心位置进行设置

2. 指令格式

```
LoadSet <抓手编号>  <工件编号>
```

<抓手编号>：0～8，对应参数 HNDDAT0～8。

<工件编号>：0～8，对应参数 WRKDAT0～8。

3. 指令例句

```
1  Oadl ON'——最佳加减速模式 = ON。
2  LoadSet 1,1'——选择 1 号抓手 HNDDAT1 及 1 号工件 WRKDAT1。
3  Mov P1'——前进到 P1 点。
4  LoadSet 0,0'——选择 0 号抓手 HNDDAT0 及 0 号工件 WRKDAT0。
5  Mov P2'——前进到 P2 点。
6  Oadl Off'——最佳加减速模式 = OFF。
```

23.11　Prec(Precision)——选择高精度模式有效或无效，用以提高轨迹精度

1. 功能

本指令选择高精度模式有效或无效，用以提高轨迹精度。

2. 指令格式

```
Prec  < On/Off >
```

Prec On——高精度模式有效。

Prec Off——高精度模式无效。

3. 指令例句

```
1  Prec On'——高精度模式有效。
```

```
2  Mvs P1'——前进到 P1 点。
3  Mvs P2'——前进到 P2 点。
4  Prec Off'——高精度模式无效。
5  Mov P1'——前进到 P1 点。
```


23.12 Torq(Torque)——转矩限制指令

1. 功能

本指令用于设置各轴的转矩限制值。

2. 指令格式

```
Torq  <轴号>  <转矩限制率>
```

<转矩限制率>：额定转矩的百分数。

3. 指令例句

```
1  Def Act 1,M_Fbd>10 GoTo *SUB1,S'——如果实际位置与指令位置差 M_Fbd 大于 10mm 则跳转到子程序*SUB1。
2  Act 1=1'——中断区间有效。
3  Torq 3,10'——设置 J3 轴的转矩限制倍率=10%。
4  Mvs P1'——前进到 P1 点。
5  Mov P2'——前进到 P2 点。
…
100  *SUB1'——程序分支标志。
101 Mov P_Fbc'——移动回当前位置。
102 M_Out(10)=1'——输出端子 10=ON。
103 End'——结束。
```

23.13 JRC(Joint Roll Change)——旋转轴坐标值转换指令

1. 功能

本指令功能是将指定旋转轴坐标值加/减 360°后转换为当前坐标值(用于原点设置或不希望当前轴受到形位标志 FLG2 的影响)。

2. 指令格式

```
JRC<[+]<数据>/-<数据>/0>[<轴号>]
```

(1) [+]<数据>——以参数 JRCQTT 设定的值为单位增加或减少的“倍数”。如果未设置参数 JRCQTT,则以 360°为单位。例如,+2 就是增加 720°,-3 就是减 1080°。

(2) 如果<数据>=0,则以参数 JRCORG 设置的值,再做原点设置(只能用于“用户定义轴”)。

(3) <轴号>——指定轴号(如果省略轴号,则为 J4 轴(水平机器人 RH-F)或 J6 轴(垂直机器人 RVH-F))。

3. 指令例句

```
Mov P1'——移动到 P1 点,J6 轴向正向旋转。
JRC +1'——将 J6 轴当前值加 360°。
Mov P1'——移动到 P1 点。
JRC +1'——将 J6 轴当前值加 360°。
Mov P1'——移动到 P1 点。
JRC -2 '——将 J6 轴当前值减 720°。
```

4. 说明

(1) 本指令只改变对象轴的坐标值,对象轴不运动(可以用于设置原点或其他用途)。

(2) 由于对象轴的坐标值改变,所以需要预先改变对象轴的动作范围,对象轴的动作范围可设置在－2340°～＋2340°。

(3) 优先轴为机器人前端的旋转轴。

(4) 未设置原点时系统会报警。

(5) 执行本指令时,机器人会停止。

(6) 使用 JRC 指令时务必设置下列参数。

① JRCEXE = 1,JRC 指令生效。

② 用参数 MEJAR 设置对象轴动作范围。

③ 用参数 JRCQTT 设置 JRC 1/－1(JRC n/－n)的动作单位。

④ 用参数 JRCORG 设置 JRC 0 时的原点位置。

23.14 Fine——定位精度

1. 功能

定位精度用脉冲数表示,即指令脉冲与反馈脉冲的差值,脉冲数越小,定位精度越高。

2. 指令格式

```
Fine  <脉冲数>,<轴号>
```

3. 说明

<脉冲数>:表示定位精度。用常数或变量设置。

<轴号>:设置轴号。

4. 程序样例

```
Fine 300'——设置定位精度为 300 脉冲。全轴通用。
Mov P1'——前进到 P1 点。
Fine 100, 2'——设置第 2 轴定位精度为 100 脉冲。
Mov P2'——前进到 P2 点。
Fine 0, 5'——定位精度设置无效。
Mov P3'——前进到 P3 点。
Fine10 0'——定位精度设置为 100 脉冲。
Mov P4'——前进到 P4 点。
```

23.15 Fine J——设置关节轴的旋转定位精度

1. 功能

本指令设置关节轴的旋转定位精度。

2. 指令格式

```
Fine  <定位精度>  J[<轴号>]
```

3. 指令例句

```
1  Fine 1, J'——设置全轴定位精度 1 [deg]。
2  Mov P1'——前进到 P1 点。
3  Fine 0.5, J, 2'——设置 2 轴定位精度 0.5 [deg]。
4  Mov P2'——前进到 P2 点。
5  Fine 0, J, 5 '——设置 5 轴定位精度无效。
6  Mov P3'——前进到 P3 点。
7  Fine 0, J'——设置全轴定位精度无效。
8  Mov P4'——前进到 P4 点。
```

23.16 Fine P——以直线距离设置定位精度

1. 功能

本指令以直线距离设置定位精度。

2. 指令格式

```
Fine  <直线距离>,P
```

3. 指令例句

```
1  Fine 1, P'——设置定位精度为直线距离 1mm。
2  Mov P1'——前进到 P1 点。
3  Fine 0, P'——定位精度无效。
4  Mov P2'——前进到 P2 点。
```

23.17 Servo(Servo)——指令伺服电源的 ON/OFF

1. 功能

本指令用于使机器人各轴的伺服 ON/OFF。

2. 指令格式

```
Servo<On/Off><机器人编号>
```

3. 指令例句

```
1  Servo On'——伺服 = ON
2   *L20:If M_Svo<>1 GoTo *L20 '——等待伺服 = ON
```

```
3  Spd M_NSpd'——设置速度
4  Mov P1'——前进到 P1 点
5  Servo Off'——伺服 = OFF
```

23.18 Reset Err(Reset Error)——报警复位

1. 功能

本指令用于使报警复位。

2. 指令格式

```
Reset Err
```

3. 指令例句

```
1  If M_Err = 1 Then Reset Err'——如果有 M_Err 报警发生,就将报警复位。
```

23.19 Wth(With)——在插补动作时附加处理的指令

1. 功能

本指令为附加处理指令。附加在插补指令之后,不能单独使用。

2. 指令例句

```
Mov P1 Wth M_Out(17) = 1 Dly M1 + 2'——在向 P1 点移动过程中指令输出端子 17 = ON, 输出端子 17 =
ON 的时间为 M1 + 2。
```

3. 说明

(1) 附加指令与插补指令同时动作。

(2) 附加指令动作的优先级如下:

```
Com > Act > WthIf(Wth)
```

23.20 WthIf(With If)——附加处理指令

在插补动作中带有附加条件的附加处理的指令。

1. 功能

本指令也是附加处理指令,只是带有"判断条件"。

2. 指令格式

```
Mov P1 WthIf  <判断条件>  <处理>
```

<处理>:处理的内容有赋值、HLT、skip。

3. 指令例句

```
Mov P1 WthIf M_In(17) = 1, Hlt'——在向 P1 点移动过程中,如果输入信号 17 = ON,则程序暂停。
```

Mvs P2 WthIf M_RSpd>200, M_Out(17)=1 Dly M1+2'——在向P2点移动过程中，如果M_RSpd>200，则指令输出端子17=ON，输出端子17=ON的时间为M1+2。

Mvs P3 WthIf M_Ratio>15, M_Out(1)=1'——在向P3点移动过程中，如果M_Ratio>15，则指令输出端子1=ON。

23.21 CavChk On——“防碰撞功能”是否生效

1. 功能

本指令用于设置“防碰撞功能”是否生效。

2. 指令格式

```
CavChk  <On/Off>[,<机器人 CPU 号>[,NOErr]]
```

<On/Off>——On：“防碰撞”停止功能＝On；Off：“防碰撞”停止功能＝Off。

<机器人 CPU 号>——设置机器人编号。

[NOErr]——检测到“干涉”时不报警。

23.22 ColLvl（ColLevel）——设置碰撞检测量级

1. 功能

本指令用于设置碰撞检测量级。

2. 指令格式

```
ColLvl  [<J1 轴>]  [<J2 轴>]  [<J3 轴>]  [<J4 轴>][<J5 轴>]  [<J6 轴>]
```

<J1～J6 轴>]：设置各轴碰撞检测量级。

3. 指令例句

```
ColLvl 80,80,80,80,80,80,,'——设置各轴碰撞检测量级。
ColChk On '——碰撞检测有效。
Mov P1'——前进到P1点。
ColLvl ,50,50,,,,, ----- '设置J2,J3轴碰撞检测量级。
Mov P2'——前进到P2点。
Dly 0.2'——暂停0.2s。
ColChk Off '——碰撞检测无效。
Mov P3'——前进到P3点。
```

23.23 Open——打开文件指令

1. 功能

本指令为“启用”某一文件指令。

2. 指令格式

```
Open  "<文件名>"  [For <模式>]  As  [#]<文件号码>
```

(1) <文件名>：记叙文件名。如果使用“通信端口”则为“通信端口名”。

(2) <模式>

INPUT——输入模式(从指定的文件里读取数据)。

OUPUT——输出模式。

APPEND——搜索模式。

“省略”——如果省略模式指定，则为搜索模式。

3. 指令例句1(通信端口类型)

```
Open "COM1:" As ＃1'——指定1＃通信口COMDEV1(传入的文件)作为＃1文件。
Mov P_01'——前进到P_01点。
Print ＃1,P_Curr'——将当前值"(100.00,200.00,300,00,400.00)(7,0)"输出到＃1文件。
Input ＃1,M1,M2,M3 '——读取＃1文件中的数据"101.00,202.00,303.00"到M1,M2,M3。
P_01.X = M1'——赋值
P_01.Y = M2'——赋值
P_01.C = Rad(M3)'——赋值
Close'——关闭所有文件。
End'——程序结束
```

4. 指令例句2(文件类型)

```
Open "temp.txt" For Append As ＃1'——将名为"temp.txt "的文件定义为＃1文件。
Print ＃1, "abc"'——在＃1文件上写"abc。
Close ＃1 '——关闭＃1文件。
```

23.24 Print——输出数据指令

1. 功能

本指令为向指定的文件输出数据。

2. 指令格式

```
Print  ＃<文件号>  <数据式1>,<数据式2>,<数据式3>
<数据式>——可以是数值表达式,位置表达式,字符串表达式
```

3. 指令例句1

```
Open "temp.txt" For APPEND As ＃1'——将temp.txt文件视作＃1文件开启。
MDATA = 150'——设置MDATA = 150。
Print ＃1," *** Print TEST *** "'——向＃1文件输出字符串" *** Print TEST *** "。
Print ＃1'——输出"换行符"。
Print ＃1,"MDATA = ",MDATA'——输出字符串"MDATA = "之后,接着输出MDATA的具体数据150。
Print ＃1'——输出"换行符"。
Print ＃1," **************** "'——输出字符串" **************************
End'——结束
```

输出结果如下。

```
 *** Print TEST ***
MDATA = 150
 ****************
```

4. 说明

(1) Print 指令后为“空白”,即表示输出换行符。注意其应用。

(2) 字符串最大为 14 字符。

(3) 多个数据以逗号分隔时,输出结果的多个数据之间有空格。

(4) 多个数据以分号分隔时,输出结果的多个数据之间无空格。

(5) 以双引号标记“字符串”。

(6) 必须输出换行符。

5. 例句 2

```
1  M1 = 123.5'——赋值
2  P1 = (130.5,-117.2,55.1,16.2,0.0,0.0)(1,0)'——赋值
3  Print #1,"OUTPUT TEST" , M1 , P1'——以逗号分隔。
```

输出结果:数据之间有空格。

```
OUTPUT TEST 123.5 (130.5,-117.2,55.1,16.2,0.0,0.0)(1,0)
```

6. 例句 3

```
3  Print #1,"OUTPUT TEST"; M1 ; P1'——以分号分隔
```

输出结果:数据之间无空格。

```
OUTPUT TEST 123.5(130.5,-117.2,55.1,16.2,0.0,0.0)(1,0)
```

7. 例句 4

在语句后面加逗号或分号,不会输出换行结果。

```
3  Print #1,"OUTPUT TEST",'——以逗号结束。
4  Print #1,M1;'——以分号结束。
5  Print #1,P1'——输出 P1 位置数据。
```

输出结果:

```
OUTPUT TEST 123.5(130.5,-117.2,55.1,16.2,0.0,0.0)(1,0)
```

23.25 Input——文件输入指令

1. 功能

从指定的文件读取“数据”的指令,读取的数据为 ASCII 码。

2. 指令格式

```
Input  #<文件编号><输入数据存放变量>[<输入数据存放变量>]…
```

(1) <文件编号>:指定被读取数据的文件号。

(2) <输入数据存放变量>:指定读取数据存放的变量名称。

3. 指令例句

```
1  Open "temp.txt" For Input As #1'——指定文件"temp.txt"为1#文件。
```

```
2  Input #1, CABC$ '——读取 1 #文件：读取时从"起首"到"换行"为止的数据被存放到变量"CABC$"。(全部为 ASCII 码。)
…
10 Close #1'——关闭 1#文件。
```

4. 说明

如果 1#文件的数据为

```
PRN MELFA,125.75,(130.5,－117.2,55.1,16.2,0,0)(1,0) CR
```

指令：1 Input #1,C1$,M1,P1

则：C1$＝MELFA

M1＝125.75

P1＝(130.5,－117.2,55.1,16.2,0,0)(1,0)

23.26　Close——关闭文件

1. 功能

将指定的文件(及通信口)关闭。

2. 指令格式

```
Close  [#]<文件号> [[#<文件号>]
```

3. 指令例句

```
1  Open "temp.txt" For Append As #1'——将文件 temp.txt 作为 1#文件打开
2  Print #1, "abc"'——在 1#文件中写入"abc"
3  Close #1  '——关闭 1#文件
```

23.27　ColChk（Col Check）——指令碰撞检测功能有效无效

1. 功能

本指令用于设置碰撞检测功能有效无效。碰撞检测功能指检测机器人手臂及抓手与周边设备是否发生碰撞,如果发生碰撞立即停止,减少损坏。

2. 指令格式

```
ColChk  On[NOErr]/Off
```

(1) On：碰撞检测功能有效。检测到碰撞发生时,立即停机,并发出 1010 报警。同时伺服＝OFF。

(2) Off：碰撞检测功能无效。

(3) NOErr：检测到碰撞发生时不报警。

3. 指令例句 1：检测到碰撞发生时报警

```
1  ColLvl 80,80,80,80,80,80,,'——设置碰撞检测量级。
```

```
2  ColChk On'——碰撞检测功能有效。
3  Mov P1'——前进到 P1 点。
4  Mov P2'——前进到 P2 点。
5  Dly 0.2'——等待动作完成。也可以使用定位精度指令 Fine。
6  ColChk Off'——碰撞检测功能无效。
7  Mov P3'——前进到 P3 点。
```

4. 指令例句 2：检测到碰撞发生时，使用中断处理

```
1   Def Act 1,M_ColSts(1) = 1 GoTo *HOME,S'——如果检测到碰撞发生,跳转到"HOME"行。
2   Act 1 = 1'——中断区间生效。
3   ColChk On,NOErr'——碰撞检测功能 = On。
4   Mov P1'——前进到 P1 点。
5   Mov P2'——前进到 P2 点。
6   Mov P3'——前进到 P3 点。
7   Mov P4'——前进到 P4 点。
8   ColChk Off'——碰撞检测功能 = Off
9   Act 1 = 0'——中断区间结束。
100 *HOME'——程序分支标志
101 ColChk Off'——碰撞检测功能 = Off
102 Servo On'——伺服 On
103 PESC = P_ColDir(1) * ( - 2)'——碰撞回退点。
104 PDST = P_Fbc(1) + PESC'——碰撞回退点计算。
105 Mvs PDST'——前进到 PDST 点。
106 Error 9100'——报警
```

5. 说明

(1) 碰撞检测是指机器人在移动过程中，实际转矩超出理论转矩达到一定量级后，则判断为碰撞，机器人紧急停止。

图 23-2 中，有理论转矩和实际检测到的转矩。如果实际检测到的转矩大于设置的转矩值，就报警。

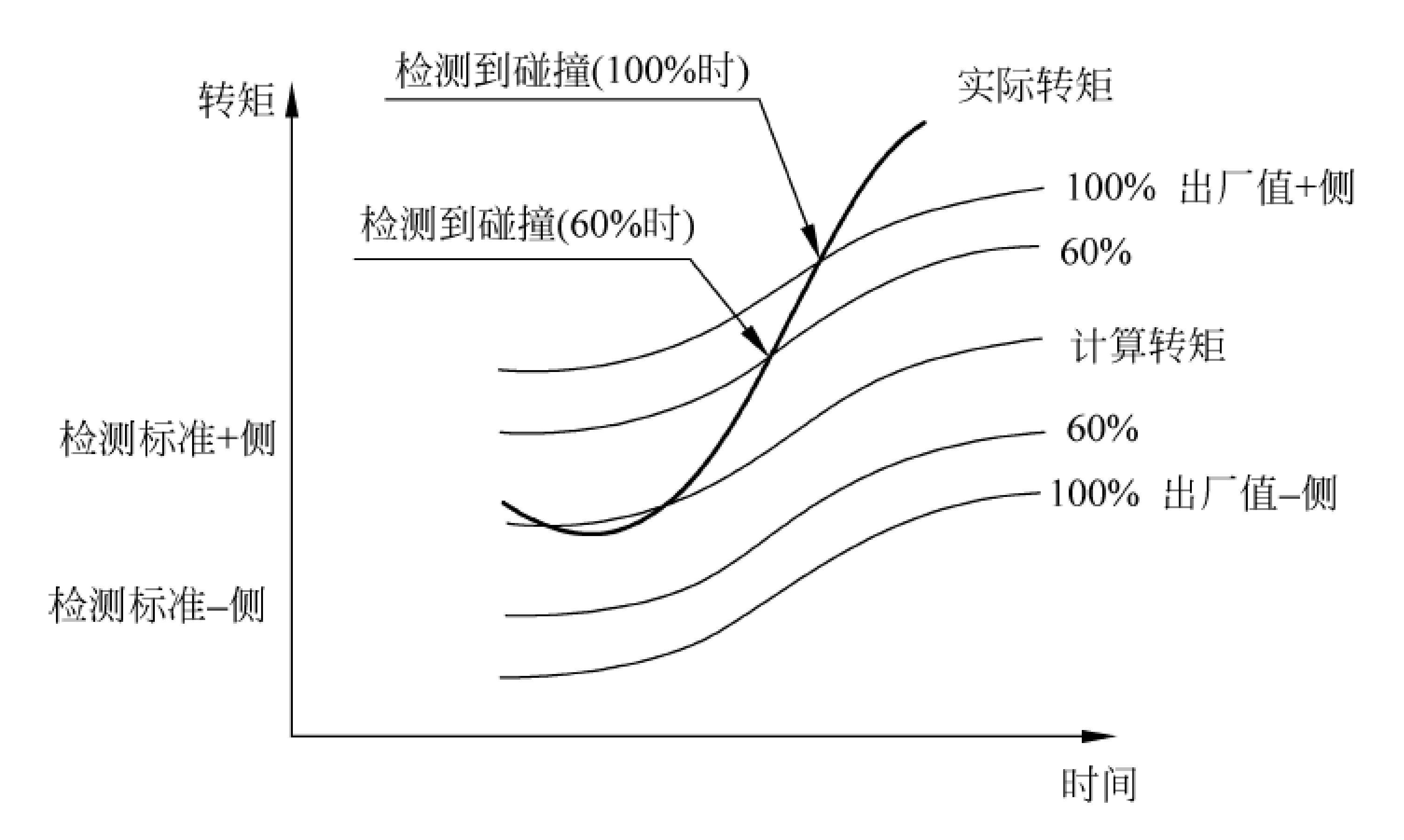


图 23-2　实际转矩与设置的检测转矩量级之间的关系

(2) 碰撞检测功能可以用参数 COL 设置。

23.28　HOpen/HClose（Open/Close）——抓手打开/关闭指令

1. 功能

本指令为抓手的 ON/OFF 指令，用于控制抓手的 ON/OFF，实质上是控制某一输出信号的 ON/OFF，所以在参数上要设置与抓手对应的输出信号。

2. 指令格式

```
HOpen  <抓手号码>
HClose  <抓手号码>
```

3. 指令例句

```
1  HOpen 1 '——指令抓手 1 = ON。
2  Dly 0.2'——暂停 0.2s。
3  HClose 1'——指令抓手 1 = OFF。
4  Dly 0.2'——暂停 0.2s。
5  Mov PUP'——前进到 PUP 点。
```

23.29　Error——发出报警信号的指令

1. 功能

本指令用于在程序中发出报警指令。

2. 指令格式

```
Error <报警编号>
```

3. 指令例句 1

```
1  Error 9000
```

4. 指令例句 2

```
4  If M1 <> 0 Then *LERR'——如果 M1 不等于 0，则跳转到 *LERR 行。
…
14  *LERR'——程序分支标志。
15 MERR = 9000 + M1 * 10'——根据 M1 计算报警号。
16 Error MERR'——报警。
17 End'——程序结束。
```

23.30　Skip——跳转指令

1. 功能

本指令的功能是中断执行当前的程序行，跳转到下一程序行。

2. 指令格式

```
Skip
```

3. 指令例句

```
1  Mov P1 WthIf M_In(17) = 1,Skip'——如果执行 MovP1 的过程中 M_In(17) = 1,则中断 MovP1 的执行,跳到下一程序行。
2  If M_SkipCq = 1 Then Hlt'——如果发生了 skip 跳转,则程序暂停。
```

23.31　Wait(Wait)——等待指令

1. 功能

本指令功能为等待条件满足后执行下一段指令。这是常用指令。

2. 指令格式

```
Wait  <数值变量> = <常数>
```

<数值变量>——数值型变量;常用的有输入输出型变量。

3. 指令例句1:信号状态

```
1  Wait M_In(1) =  1'——与 *L10:If M_In(1) = 0 Then GoTo *L10 功能相同。
2  Wait M_In(3) = 0'——等待输入端子 3 = OFF。
```

4. 指令例句2:多任务区状态

```
3  Wait M_Run(2) = 1 '——等待任务区 2 程序启动。
```

5. 指令例句3:变量状态

```
Wait M_01 = 100   '——如果变量"M_01 = 100",就执行下一行。
```

23.32　Clr(Clear)——清零指令

1. 功能

本指令用于对输出信号、局部变量、外部变量及数据"清零"。

2. 指令格式

```
Clr  < TYPE >
```

< TYPE >——清零类型。

< TYPE >=1:输出信号复位。

< TYPE >=2:局部变量及数组清零。

< TYPE >=3:外部变量及数组清零。但公共变量不清零。

3. 指令例句1:类型1

```
Clr 1'——将输出信号复位。
```

4. 指令例句2:类型2

```
Dim MA(10)'——定义数组。
```

```
Def Inte IVAL'——定义变量精度。
Clr 2'——MA(1)～MA(10)及变量 IVA 及程序内局部变量清零。
```

5. 指令例句 3：类型 3

```
Clr 3'——外部变量及数组清零。
```

6. 指令例句 4：类型 0

```
Clr 0'——同时执行类型 1～3 清零。
```

23.33　END——程序段结束指令

1. 功能

END 指令在主程序内表示程序结束。在子程序内表示子程序结束并返回主程序。

2. 指令格式

```
END
```

3. 指令例句

```
1  Mov P1'——前进到 P1 点。
2   GoSub * ABC'——调用子程序。
3   End'——主程序结束。
…
10   * ABC'——程序分支标志。
11  M1 = 1'——赋值。
12   Return'——返回。
```

4. 说明

(1) 如果需要程序中途停止并处于中断状态，应该使用 HLT 指令。

(2) 可以在程序中多处编制 END 指令。也可以在程序的结束处不编制 END 指令。

23.34　For Next——循环指令

1. 功能

本指令为循环指令。

2. 指令格式

```
For <计数器> = <初始值> To <结束值> Step <增量>
Next  <计数器>
```

(1) <计数器>——循环判断条件。

(2) Step <增量>——每次循环增加的数值。

(3) 指令例句：求 1～10 的和。

```
1  MSUM = 0 '——设置"MSUM = 0"
2  For M1 = 1 To 10'——设置 M1 从 1 到 10 为循环条件,单步增量 = 1
3  MSUM = MSUM + M1'——计算公式
4  Next M1
```

4. 说明

(1) 循环嵌套为 16 级。

(2) 跳出循环不能使用 GOTO 语句,使用 LOOP 语句。

23.35 Return——子程序/中断程序结束及返回

1. 功能

本指令是子程序结束及返回指令。

2. 指令格式

```
Return ——子程序结束及返回。
Return<返回程序行指定方式>
<返回程序行指定方式>——0: 返回到中断发生的"程序步"。
                    ——1: 返回到中断发生的"程序步"的下一步。
```

3. 指令例句 1: 子程序调用

```
1    ' *** MAIN PROGRAM ***
2    GoSub * SUB_INIT'——跳转到子程序 * SUB_INIT 行
3    Mov P1'——前进到 P1 点
…
100  ' *** SUB INIT *** '
101  * SUB_INIT '——子程序标记
102  PSTART = P1'——设置
103  M100 = 123'——赋值
104  Return 1'——返回到"子程序调用指令"的下一行(即第 3 步)
```

4. 指令例句 2: 中断程序调用

```
1   Def Act 1,M_In(17) = 1 GoSub * Lact '——定义 Act 1 对应的中断程序
2   Act 1 = 1'——中断区间生效
…
10  * Lact'——程序分支标志
11  Act 1 = 0'——中断区间结束
12  M_Timer(1) = 0'——赋值
13  Mov P2'——前进到 P2 点
14  Wait M_In(17) = 0'——等待
15  Act 1 = 1'——中断区间生效
16  Return 0 '——返回到发生"中断"的单步
```

5. 说明

以 GoSub 指令调用子程序,必须以 Return 作为子程序的结束。

23.36　Label——标签、指针

1. 功能

“标签”用于为程序的分支处做标记，属于程序结构流程用标记。

2. 指令例句

```
1  * SUB1 '—— * SUB1 即是"标签"。
2  If M1 = 1 Then GoTo * SUB1 '——判断语句
3  * LBL1:If M_In(19) = 0 Then GoTo * LBL1 '——判断语句
* LBL1 即是标签。
```

23.37　Tool(Tool)——Tool 数据的指令

1. 功能

本指令用于设置 Tool 的数据，适用于双抓手的场合，Tool 数据包括抓手长度、机械 I/F 位置、形位。

2. 指令格式

```
Tool  < Tool 数据>
```

< Tool 数据>——以位置点表达的 Tool 数据。

3. 指令例句 1

直接以数据设置。

```
1  Tool (100,0,100,0,0,0)'——设置一个新的 Tool 坐标系。新坐标系原点 X = 100mm,Z  = 100mm。
(实际上变更了"控制点"。)
2  Mvs P1'——前进到 P1 点。
3  Tool P_NTool'——返回初始值(机械 IF,法兰面)。
```

4. 指令例句 2

以直角坐标系内的位置点设置。

```
1  Tool PTL01'——设置一个新的 Tool 坐标系。以 PTL01 为原点。
2  Mvs P1'——前进到 P1 点。
```

如果 PTL01 位置坐标为(100，0，100，0，0，0，0，0)，则与指令例句 1 相同。

5. 说明

(1) 本指令适用于双抓手的场合。每个抓手的控制点不同。单抓手的情况下一般使用参数 MEXTL 设置即可。

(2) 使用 Tool 指令设置的数据存储在参数 MEXTL 中。

(3) 可以使用变量 M_Tool，将 METL1～4 设置到 Tool 数据中。

23.38 Base——设置一个新的“世界坐标系”

1. 功能

本指令通过设置偏置坐标建立一个新的世界坐标系。偏置坐标为以世界坐标系为基准观察到的基本坐标系原点的坐标值，参看图 23-3。

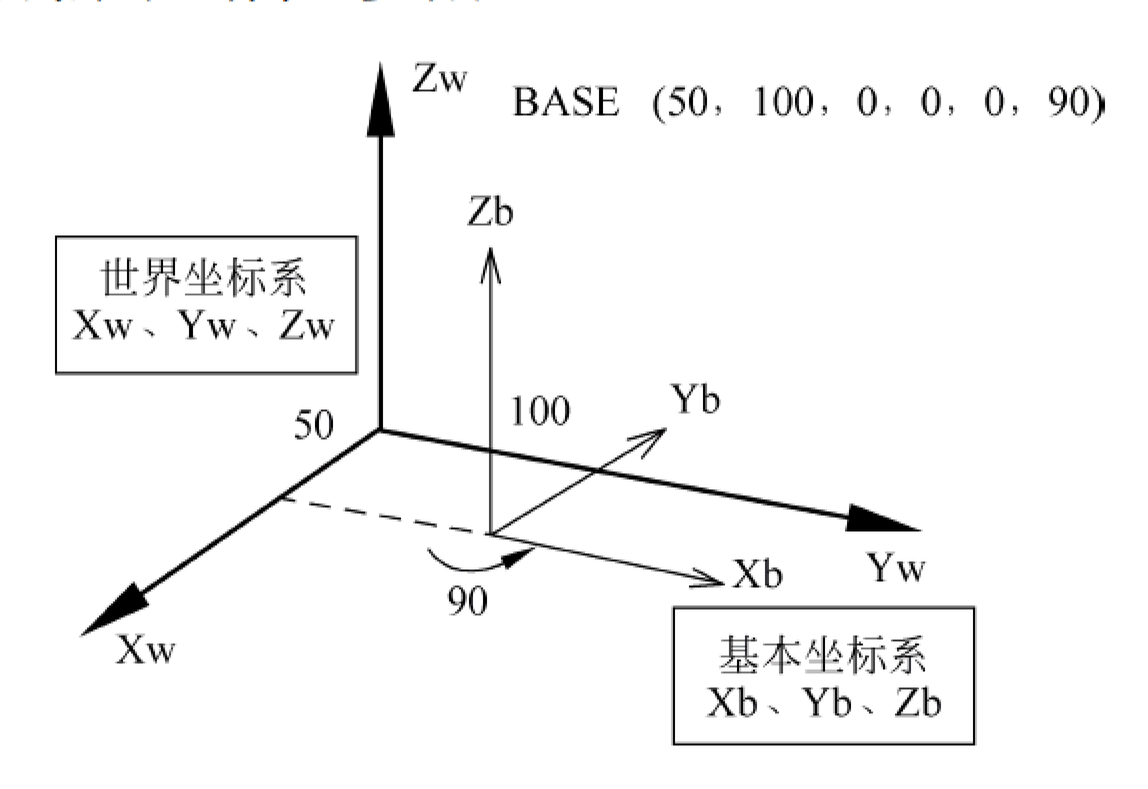


图 23-3 世界坐标系与基本坐标系的关系

2. 指令格式

Base <新原点>——用新原点表示一个新的世界坐标系。

Base <坐标系编号>——用坐标系编号选择一个新的世界坐标系。

0：系统初始坐标系 P_NBase。P_NBase=0，0，0，0，0，0。

1～8：工件坐标系 1～8。

3. 指令例句 1

1 Base (50,100,0,0,0,90)'——以新原点设置一个新的世界坐标系。这个点是基本坐标系原点在新坐标系内的坐标值。

2 Mvs P1'——前进到 P1 点。

3 Base P2'——以 P2 点为基点设置一个新的世界坐标系。

4 Mvs P1'——前进到 P1 点。

5 Base 0 '——返回初始世界坐标系。

4. 指令例句 2：以坐标系编号选择坐标系

1 Base 1'——前进到 P1 点。选择 1 号坐标系 WK1CORD。

2 Mvs P1'——前进到 P1 点。

3 Base 2'——前进到 P1 点。选择 2 号坐标系 WK2CORD。

4 Mvs P1'——前进到 P1 点。

5 Base 0 '——前进到 P1 点。选择初始世界坐标系。

5. 说明

(1) 新原点数据是从新世界坐标系观察到的基本坐标系原点的位置数据，即基本坐标系在新世界坐标系中的位置。

(2) 使用当前位置点建立一新世界坐标系时可以使用 Base Inv(P1)指令(必须对 P1 点进行逆变换)。

23.39 XLoad(X Load)——加载程序指令

1. 功能

加载程序。多程序时,选择任务区并加载程序号。

2. 指令格式

```
XLoad  <任务区号>  <程序号>
```

3. 指令例句

```
If M_Psa(2) = 0 Then ＊LblRun'——判断执行语句。
XLoad 2,"10"'——在任务区 2 加载 10 号程序。
＊L30:If C_Prg(2)<>"10" Then GoTo ＊L30'——判断执行语句。
XRun 2'——任务区 2 启动运行。
Wait M_Run(2) = 1'——等待。
＊LblRun'——程序分支标志。
```

23.40 XRun(X Run)——多任务工作时的程序启动指令

1. 功能

本指令用于在多任务工作时指定“任务区号”和“程序号”及“运行模式”。

2. 指令格式

```
XRun  <任务区号>  "<程序名>"  <运行模式>
    <运行模式>——设置程序"连续运行"或"单次运行"。
    <运行模式> = 0: 连续运行。
    <运行模式> = 1: 单次运行。
```

3. 指令例句 1

```
XRun 2,"1"'——指令运行任务区 2 内的 1 号程序。连续运行模式。
Wait M_Run(2) = 1'——等待运行任务区 2 内的 1 号程序启动完成。
```

4. 指令例句 2

```
XRun  3,"2",1'——指令运行任务区 3 内的 2 号程序。单次运行模式。
Wait M_Run(3) = 1'——等待运行任务区 3 内程序启动完成。
```

5. 指令例句 3

```
XLoad 2, "1"'——在任务区 2 内加载＃1 程序。
＊LBL: If C_Prg(2)<>"1" Then GoTo ＊LBL'——等待加载完毕。
XRun 2'——指令运行任务区 2 内程序。
```

6. 指令例句 4

```
1  XLoad 3, "2"'——在任务区 3 内加载#2 程序。
2  *LBL:If C_Prg (3)<>"2" Then GoTo *LBL'——等待加载完毕。
3  XRun 3,,1 '——指令运行任务区 3 内程序单次运行模式。
```

本指令中,“程序名”必须要用双引号。

23.41 XStp(X Stop)——多任务工作时的程序停止指令

1. 功能

本指令为多任务工作时的程序停止指令,需要指定任务区号。

2. 指令格式

```
XStp  <任务区号>
```

3. 指令例句

```
1  XRun 2'——指令运行任务区 2 内的程序
10 XStp 2'——任务区 2 内的程序停止。
11 Wait M_Wai(2) = 1'——等待。
20 XRun 2'——指令运行任务区 2 内的程序。
```

23.42 XRst(X Reset)——复位指令

1. 功能

程序复位指令,用于多任务工作时指令某一任务区程序的复位。

2. 指令格式

```
XRst  <任务区号>
```

3. 指令例句

```
1  XRun 2'——指令任务区 2 启动。
2  Wait M_Run(2) = 1'——等待任务区 2 启动完成。
10 XStp 2 '——指令任务区 2 停止。
11 Wait M_Wai(2) = 1'——等待任务区 2 停止完成。
…
15 XRst 2'——指令任务区 2 内的程序复位。
16 Wait M_Psa(2) = 1'——等待任务区 2 内的程序复位完成。
…
20 XRun 2'——指令任务区 2 启动。
21 Wait M_Run(2) = 1'——等待任务区 2 启动完成。
```

本指令必须在程序暂停状态下执行,在其他状态下执行会报警。

23.43　XClr(X Clear)——多程序工作时，解除某任务区的程序选择状态

1. 功能

多程序工作时，解除某任务区的程序选择状态，使该任务区处于可以重新加载程序的状态。

2. 指令格式

```
XClr  <任务区号>
```

3. 指令例句

```
1  XRun 2,"1"'——运行任务区 2 内的 1 号程序。
10 XStp 2'——停止任务区 2 运行。
11 Wait M_Wai(2) = 1'——等待任务区 2 中断启动。
12 XRst 2'——解除任务区 2 程序中断状态。
13 XClr 2'——解除任务区 2 程序选择状态。
14 End'——程序结束。
```

23.44　GetM (Get Mechanism)——指定获取机器人控制权

1. 功能

本指令用于指定机器人的控制权。在多任务控制时，在任务区(插槽)1 以外的程序要执行对机器人控制，或对附加轴作为用户设备控制时，使用本指令。

2. 指令格式

```
GetM   <机器人编号>
   <机器人编号>——使用的机器人编号。
```

3. 指令例句 1

```
1  RelM'——解除"机器人控制权"。这样可以从任务区 2 对机器人 1 的任务区 1 程序进行控制。
2  XRun 2,"10"'——在任务区 2 选择并运行 10 号程序。
3  Wait M_Run(2) = 1'——等待任务区 2 的程序启动。
```

任务区 2 内的"10"号程序：

```
1  GetM 1'——取得 1 号机器人的控制权。
2  Servo On'—— 1 号机器人伺服 On。
3  Mov P1'——前进到 P1 点。
4  Mvs P2'——前进到 P2 点。
5  P3 = P_Curr'——设置 P3 点为当前位置。
6  Servo Off'——1 号机器人伺服 Off。
7  RelM'——解除对 1 号机器人的控制权。
8  End'——程序结束。
```

4. 说明

(1) 一般执行单任务时在初始状态就获得对机器人1的控制权，所以不使用本指令。

(2) 不能使多个程序同时获得对机器人1的控制权，所以对于任务区1以外的程序，要对机器人1进行控制必须按以下步骤执行。

① 在任务区1的程序中，解除对机器人1的控制权。

② 在其他任务区的程序中，使用GetM 1获得对机器人1的控制权。

已经获得对机器人1的控制权的程序中，再发GetM 1指令会报警。

23.45 RelM(Release Mechanism)——解除机器控制权

1. 功能

在多任务工作时，为了从其他任务区(插槽)对任务区1进行控制，需要解除任务区1的控制权。本指令就是解除控制权指令。

2. 指令格式

```
Relm
```

3. 指令例句

先在任务区1内解除控制权，再运行任务区2的程序，从任务区2对任务区1的程序进行控制。

```
RelM'——解除任务区1的控制权。
XRun 2,"10"'——指令任务区2内运行10号程序。
Wait M_Run(2) = 1'——等待任务区2程序启动。
```

任务区2内的是"10"号程序：

```
GetM 1 '——获取任务区1控制权。
Servo On '——伺服On。
Mov P1'——前进到P1点。
Mvs P2'——前进到P2点。
Servo Off'——伺服Off。
RelM'——解除对任务区1的控制权。
End'——程序结束。
```

23.46 Priority(Priority)——优先执行指令

1. 功能

本指令在多任务时使用，指定各任务区(插槽)内程序的执行行数。

2. 指令格式

```
Priority  <执行行数> [<任务区号>]
```

<执行行数>——设置执行程序的行数。

<任务区号>——任务区号。

3. 指令例句

任务区 1

10 Priority 3'——指定执行任务区 1 内的程序 3 行。(如果省略任务区号,就是指当前任务区。)

任务区 2

10 Priority 4'——指定执行任务区 2 内的程序 4 行。(如果省略任务区号,就是指当前任务区。)

动作:先执行任务区 1 内程序 3 行,再执行任务区 2 内程序 4 行,循环执行。

23.47 需要思考的问题

(1) 如何执行机器人柔性控制?

(2) 如何执行机器人转矩控制?

(3) 如何执行 input 指令?

(4) 如何执行 SKIP 指令?

(5) 如何执行循环指令?

(6) label 指令有什么作用?

第24章 第24日——触摸屏在机器人上的应用

【学习目的】

在实际工作中，机器人不是单一的在工作，往往需要与PLC、触摸屏、上位机一同工作。机器人与触摸屏联合使用的方法有两种，第一种是触摸屏连接在PLC上，PLC再与机器人相连。第二种是触摸屏直接与机器人相连。第二种方法能够在触摸屏上直接读取机器人各种工作状态变量，同时也可以节省一个PLC。本章学习触摸屏直接与机器人相连的方法。

24.1 概　　述

触摸屏（以下简称GOT）可以与机器人通过以太网直接相连。通过GOT可以直接控制机器人的启动、停止，选择程序号，设置速度倍率，监视机器人的工作状态，执行JOG操作等。

本章以三菱GOT与机器人的连接为例，介绍GOT界面的制作过程以及与之相应的机器人一侧的参数设置。在GOT界面制作中使用MELSOFT GT Design 3软件（简称GT3）。在机器人一侧使用RT ToolBox2软件（简称RT），这些软件在三菱官网上都可下载。在下文中不再重复提及。

24.2 GOT与机器人控制器的连接及通信参数设置

24.2.1 GOT与机器人控制器的连接

如图24-1所示，GOT与机器人控制器可以通过以太网直接连接。

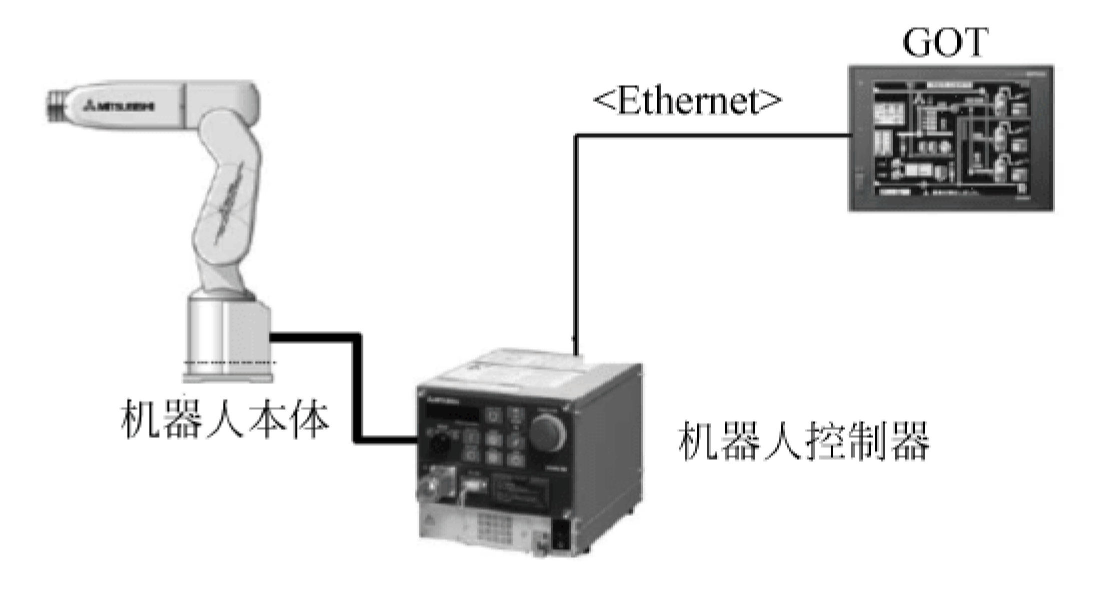


图24-1　GOT与机器人控制器的连接

24.2.2 GOT 机种选择

使用 GT 3 软件，在图 24-2 中，选择 GS 系列 GOT。GS 系列 GOT 是最经济型的 GOT。



图 24-2 GOT 类型型号及语言设置

24.2.3 GOT 一侧通信参数设置

图 24-3 是 GOT 自动默认的以太网通信参数。

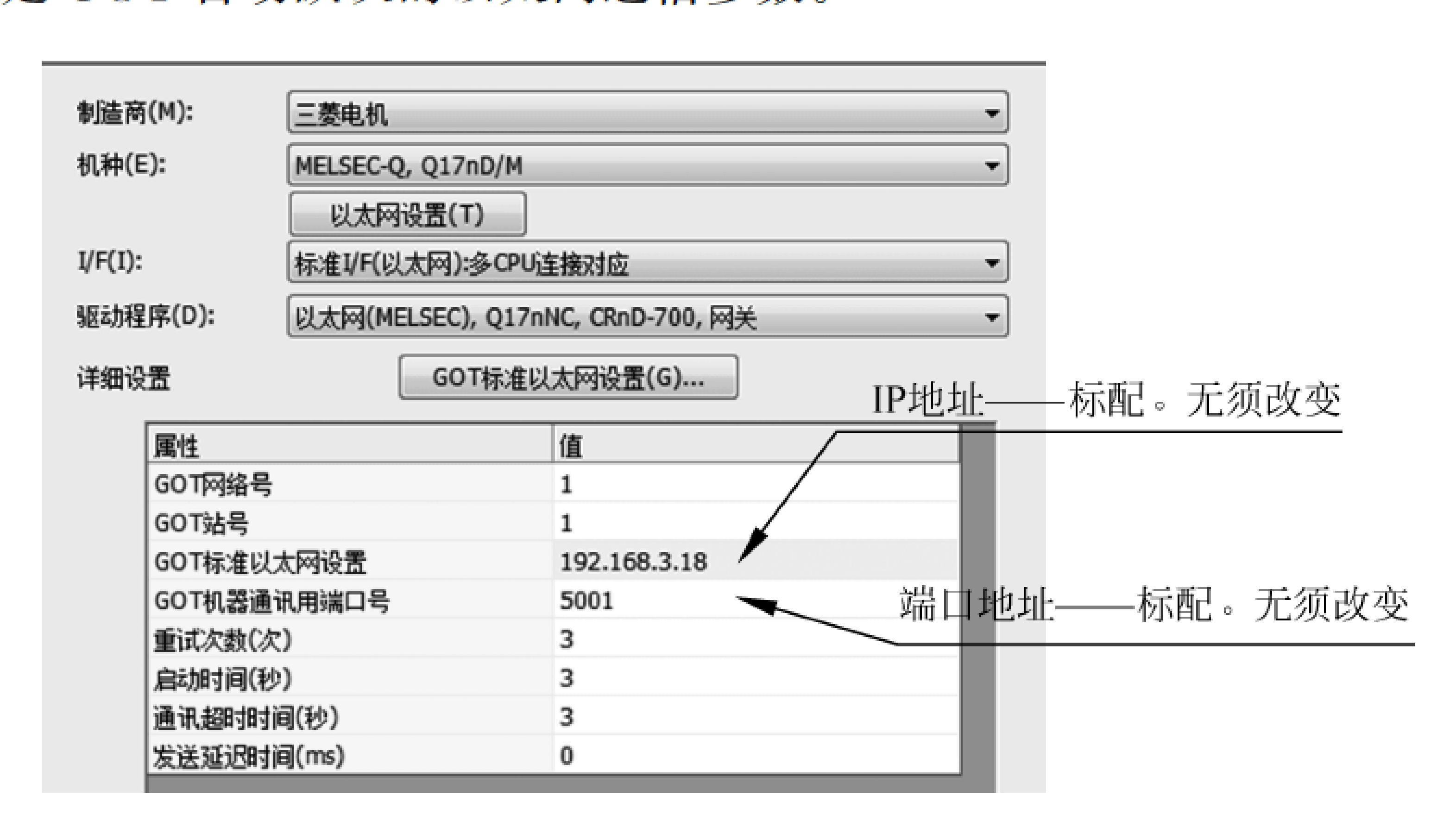


图 24-3 GOT 自动默认的以太网通信参数

在图 24-4 下方，是对 GOT 所连接对象(机器人)一侧通信参数的设置。请按照图 24-4 进行设置(注意这是在 GOT 软件上的设置)，方法如下。

(1) IP 地址必须与 GOT 在同一网段，但是第 4 位数字必须不同。

(2) 必须设置“站号＝2”。

(3) 设置“端口号＝5001”。在选择 GOT 所连接的机器类型后，“端口号”自动改变。

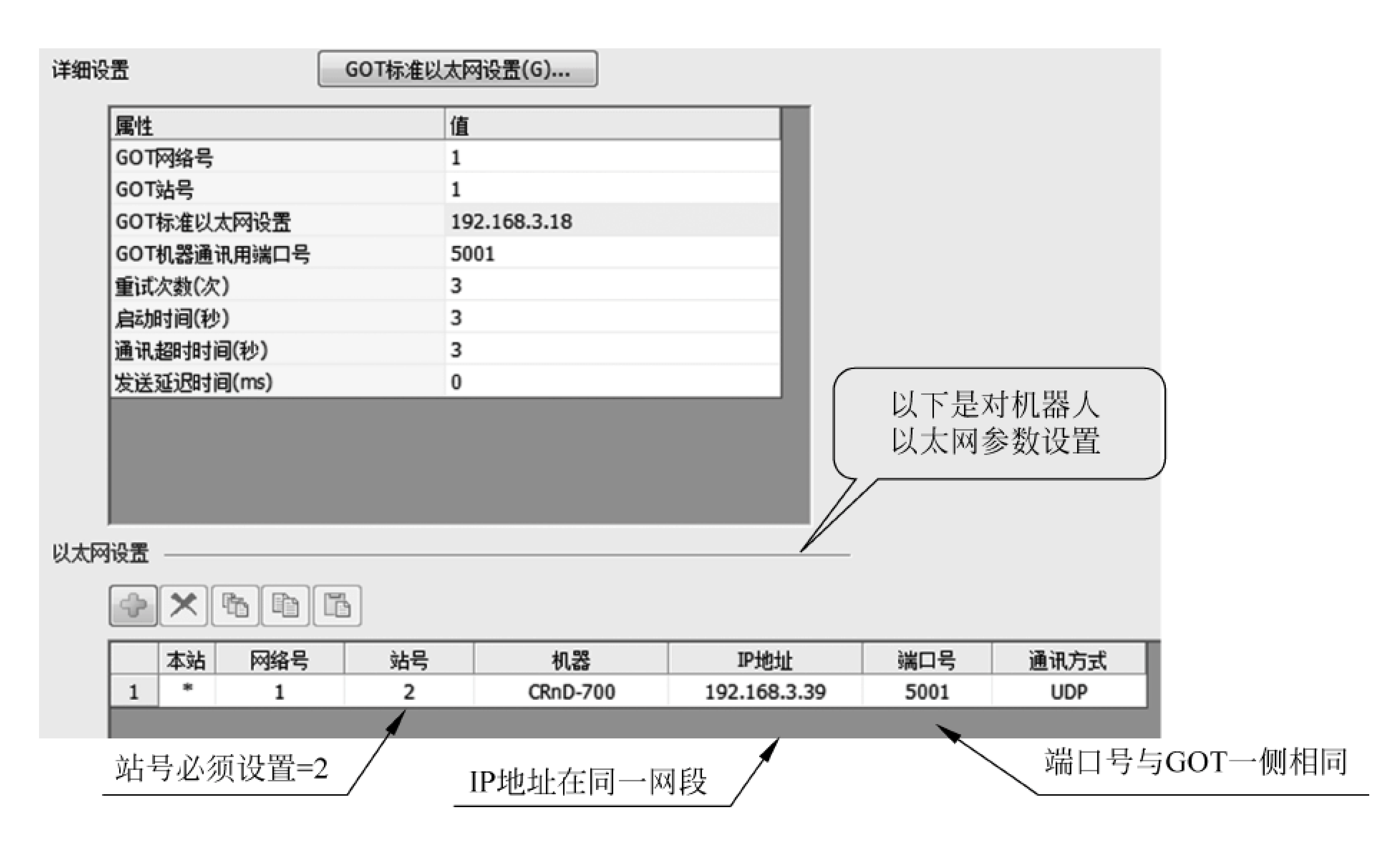


图 24-4　从 GOT 软件对机器人一侧以太网通信参数的设置

24.2.4　机器人一侧通信参数的设置

打开机器人软件 RT ToolBox2，设置 Ethernet 通信参数。

(1) 单击“任务区”→“参数”→“通信参数”→“Ethernet 设置”，如图 24-5 所示。

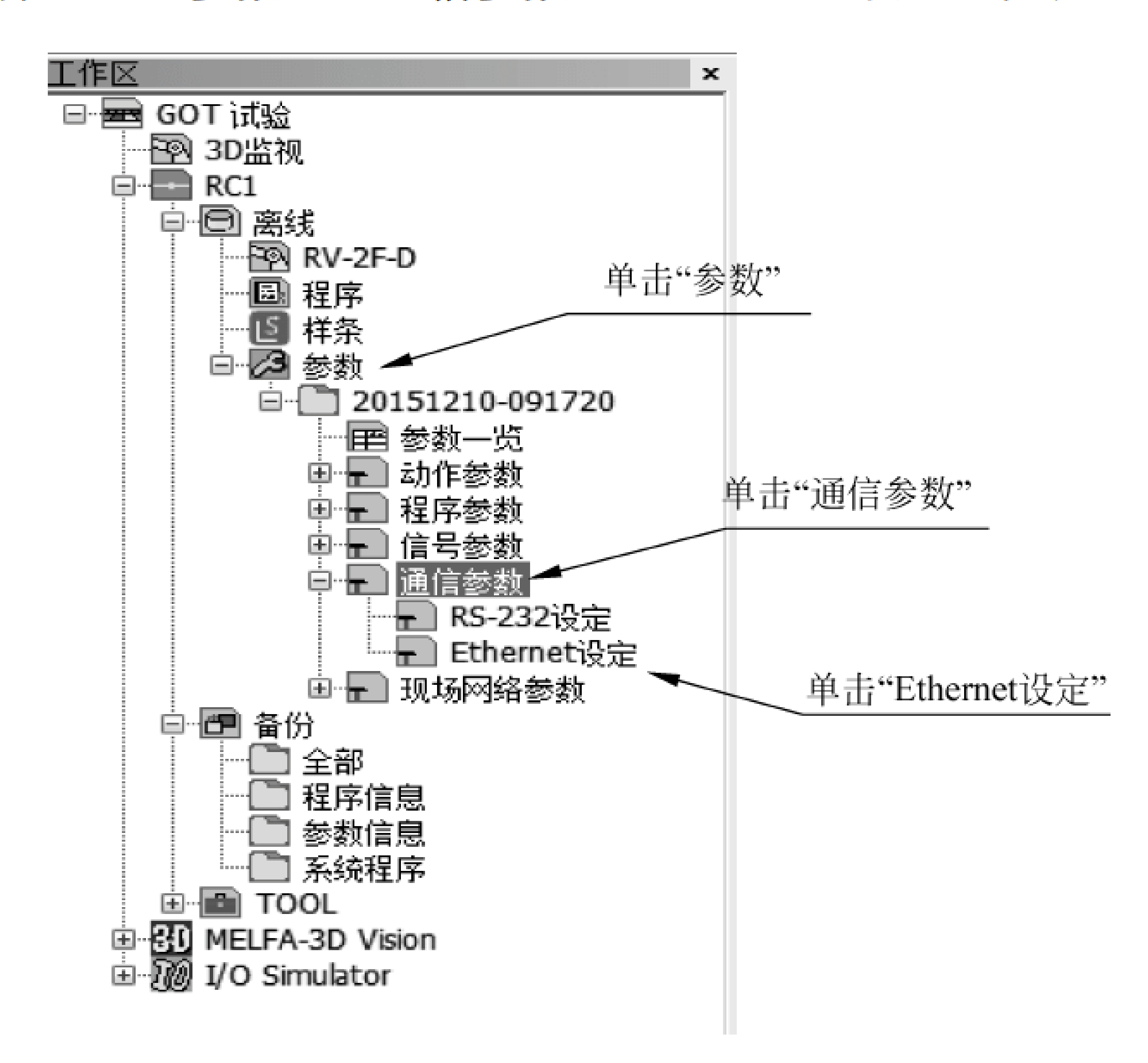


图 24-5　打开机器人以太网参数设置界面

(2) 在弹出的界面上，进行 IP 地址设置，本 IP 地址是机器人一侧的 IP。本设置必须与图 24-6 相同。设置原则也是 IP 地址必须与 GOT 在同一网段，但是第 4 位数字必须不同。

(3) 设备・端口设置。

单击“设备・端口”，弹出如图 24-7 所示对话框。



图 24-6　机器人以太网参数设置界面

设备是指与机器人连接的设备，即 GOT。其设置就是对 GOT 通信参数设置，应该按 GOT 一侧的标准参数设置，如图 24-7 所示。

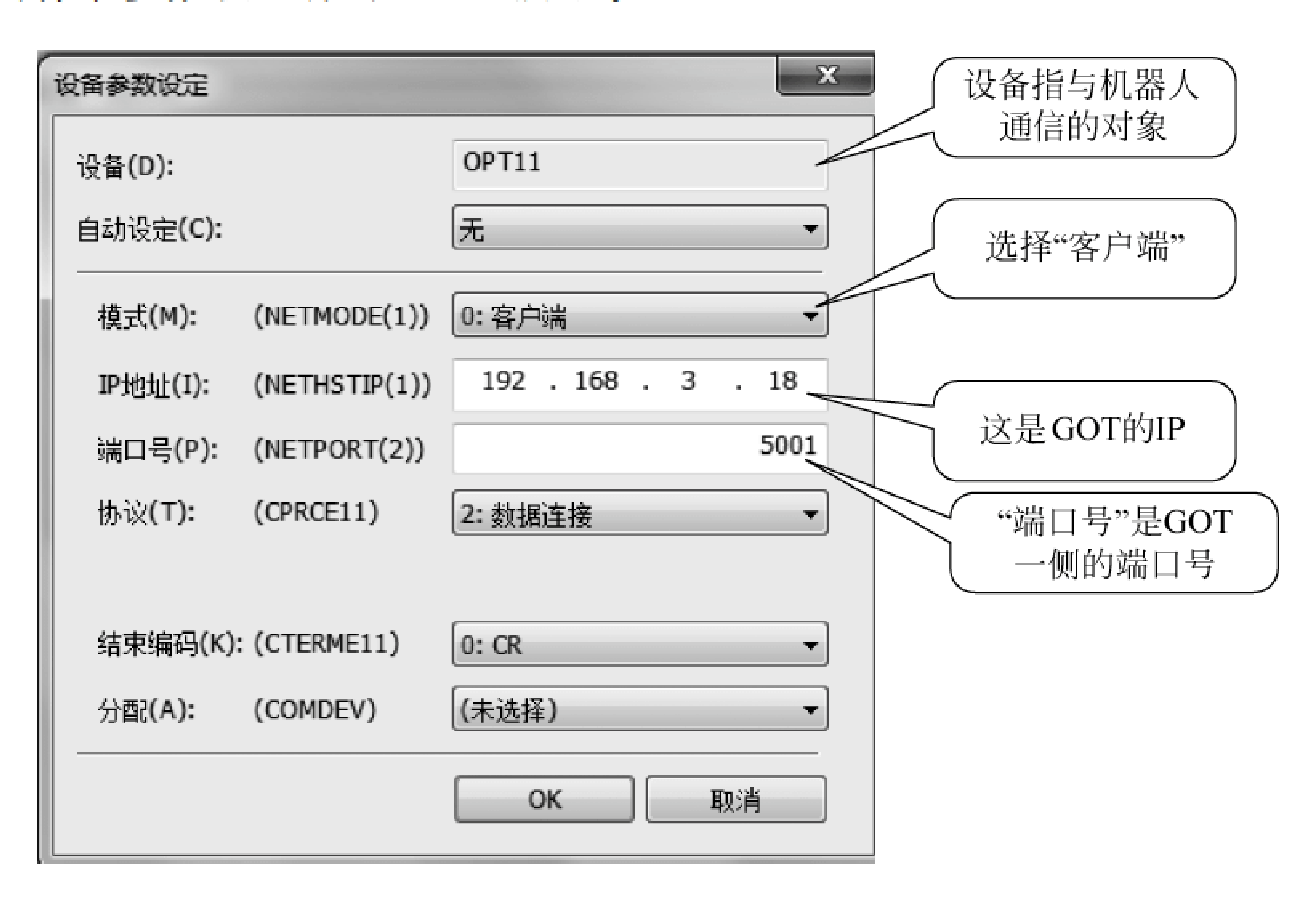


图 24-7　在机器人一侧对 GOT 以太网参数的设置画面(设备参数设定)

设置方法如下。

① 设备名称：OPT11。

② 模式(NETMODE(1))：选择“客户端”。

③ IP 地址：为 GOT 一侧 IP。

④ 端口号：GOT 一侧使用的端口。

经过在 GOT 一侧及机器人一侧做通信参数的设置，连上以太网线后，就可以进行通信了。

24.3　输入输出信号界面制作

图 24-8 是在 GOT 上制作的“操作屏”界面，该界面上的各按键都是“开关型”按键，对应机器人内部的“输入输出信号”。



图 24-8　操作屏界面

24.3.1　GOT器件与机器人I/O地址的对应关系

在机器人一侧，其输入信号为10000～18191、输出信号为10000～18191，用于与GOT通信使用。

在GOT一侧，器件U3E0-10000～10511用作输入器件。器件U3E1-10000～10511用作输出器件。其对应关系如表24-1和表24-2所示。

在表24-1中，机器人输入信号"对应的功能"是推荐使用(用户可自行设置)，这些功能还需要在机器人一侧通过参数进行设置。

表 24-1　输入信号对应表

序　　号	GOT	机器人(int)	推荐对应的功能信号	
			功能	参数名称
	U3E0-10000～U3E0-10511	10000～18191	输入点范围	
1	U3E0-10000.b0	10000		
2	U3E0-10000.b1	10001		
3	U3E0-10000.b2	10002		
4	U3E0-10000.b5	10005	操作权	IOENA
5	U3E0-10000.b6	10006	启动	START
6	U3E0-10000.b8	10008	程序复位	SLOTINIT
7	U3E0-10000.b9	10009	报警复位	ERRRESET
8	U3E0-10000.b10	10010	伺服ON	SRVON
9	U3E0-10000.b11	10011	伺服OFF	SRVOFF
10	U3E0-10000.b12	10012	程序循环结束	CYCLE
11	U3E0-10000.b13	10013	回退避点	SAFEPOS
12	U3E0-10000.b15	10015	全部输出信号复位	OUTRESET
13	U3E0-10001.b4	10020	选定程序号确认	PRGSEL
14	U3E0-10001.b5	10021	选定速度倍率确认	OVRDSEL
15	U3E0-10001.b6	10022	指令输出"程序号"	PRGOUT
16	U3E0-10001.b7	10023	指令输出"程序行号"	LINEOUT
17	U3E0-10001.b8	10024	指令输出"速度倍率"	OVRDOUT
18	U3E0-10001.b9	10025	指令输出"报警号"	ERROUT
18	U3E0-10002	10032～10047	数据输入区	IODATA

表 24-2　输出信号对应表

序　　号	GOT	机器人(out)	推荐对应的功能信号	
			功能	参数名称
	U3E1-10000～U3E1-10511	10000～18191		
1	U3E1-10000. b0	10000	暂停状态	STOP2
2	U3E1-10000. b1	10001	控制器上电 ON	RCREADY
3	U3E1-10000. b2	10002	远程模式状态	ATEXTMD
4	U3E1-10000. b3	10003	示教模式状态	TEACHMD
5	U3E1-10000. b4	10004	示教模式状态	ATTOPMD
6	U3E1-10000. b5	10005	操作权＝ON	IOENA
7	U3E1-10000. b6	10006	自动启动＝ON	START
8	U3E1-10000. b7	10007	STOP＝ON	STOPSTS
9	U3E1-10000. b8	10008	可重新选择程序	SLOTINIT
10	U3E1-10000. b9	10009	发生报警	Error occurring output
11	U3E1-10000. b10	10010	伺服 ON 状态	SRVON
12	U3E1-10000. b11	10011	伺服 OFF 状态	SRVOFF
13	U3E1-10000. b12	10012	循环停止状态	CYCLE
14	U3E1-10000. b13	10013	退避点返回状态	SAFEPOS
15	U3E1-10000. b14	10014	电池电压过低状态	BATERR
16	U3E1-10001. b1	10016	H 级报警状态	HLVLERR
17	U3E1-10001. b2	10017	L 级报警状态	LLVLERR
18	U3E1-10001. b6	10022	程序号输出状态	PRGOUT
19	U3E1-10001. b7	10023	程序行号输出状态	LINEOUT
20	U3E1-10001. b8	10024	倍率输出状态	OVRDOUT
21	U3E1-10001. b9	10025	报警号输出状态	ERROUT
22	U3E1-10002	10032～10047	数据输出地址	

24.3.2　输入输出点器件制作方法

以自动启动按键为例,说明输入输出点制作方法。

制作“位开关型”按键——“启动”按键。

(1) 在 GOT 界面上制作按键,如图 24-9 所示。

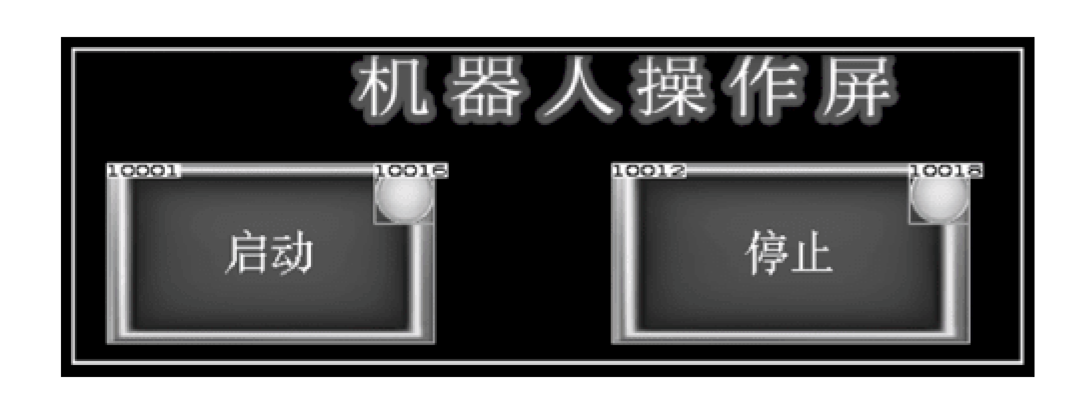


图 24-9　GOT 界面制作

(2) 设置“启动”按键的器件号为 U3E0-10000. b6。该按键 U3E0-10000. b6,如图 24-10 所示对应到机器人中的输入信号地址为 10006,见表 24-1。

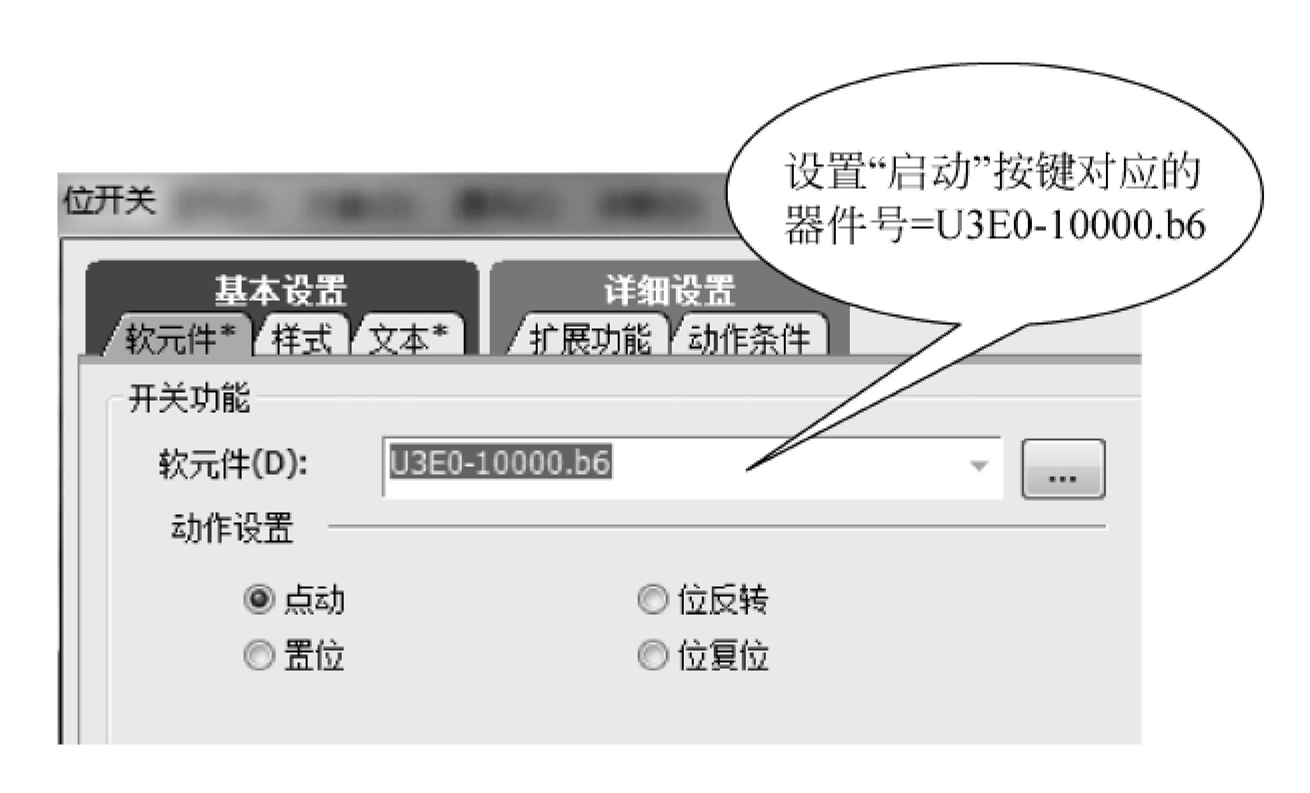


图 24-10 “启动”按键器件号的设置

(3) 在机器人软件 RT ToolBox 中设置“启动”功能对应的输入地址号为“10006”，如图 24-11 所示。



图 24-11 “启动”功能对应的输入地址号为“10006”

经过以上编写设置，触摸屏上的“启动”按键就对应了机器人的“启动”功能。

24.4 程序号的设置与显示

程序号的设置与显示是客户最基本的要求之一。其制作方法如下，参见图 24-12。

24.4.1 程序号的选择设置

(1) 在 GOT 上制作一“数据输入”器件，该器件用于输入数据。器件的地址号根据表 24-1 设置为“U3E0-10002”，如图 24-13 所示。

(2) 在机器人参数中，IODATA 参数用于设置“数据输入”的一组“输入信号的起始”地址。将与 GOT 器件 U3E0-10002 对应的输入地址 10032～10047 设置到参数 IODATA 中，如图 24-14 所示。

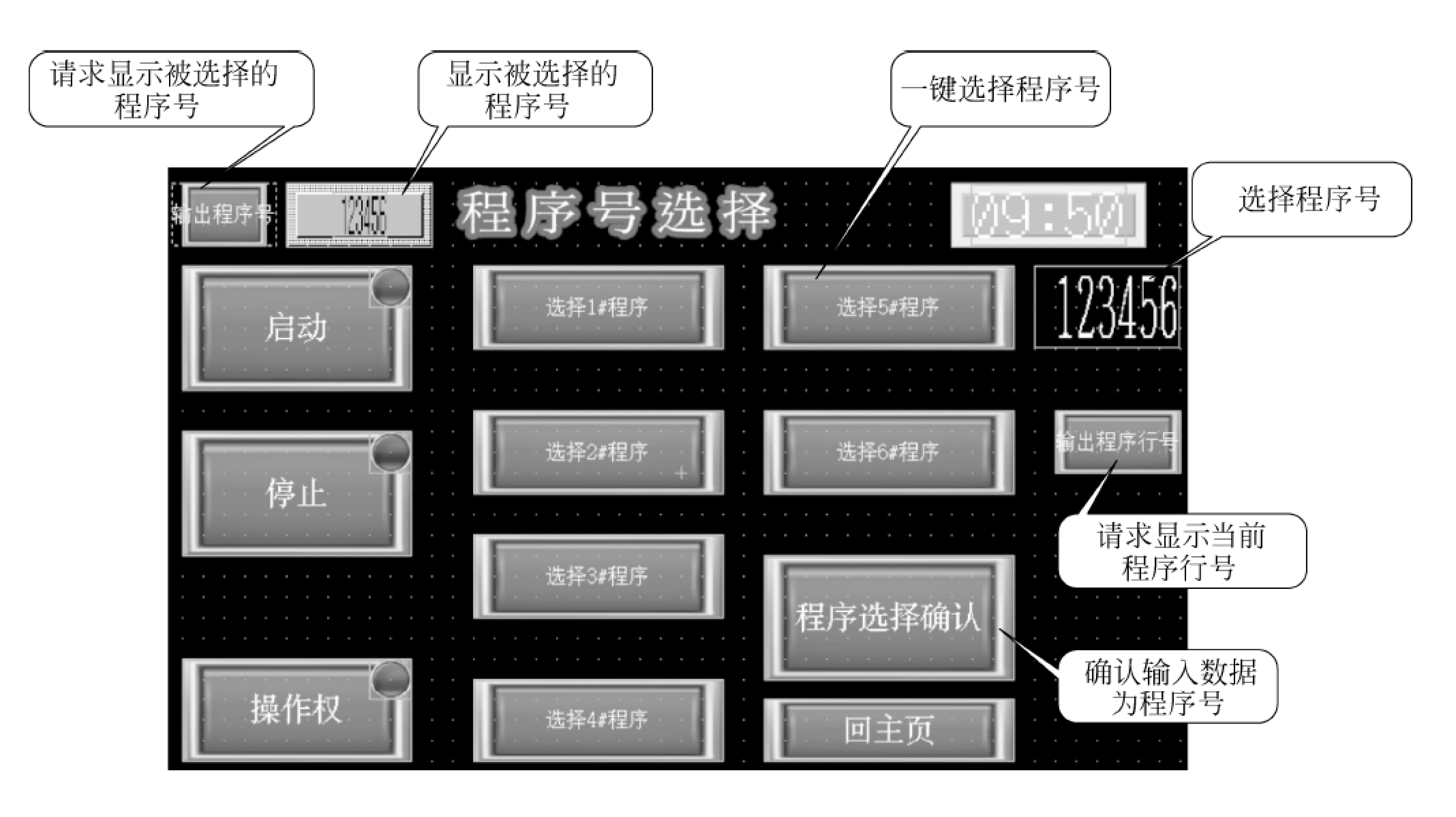


图 24-12　程序号选择屏的制作与设置



图 24-13　器件地址号的设置

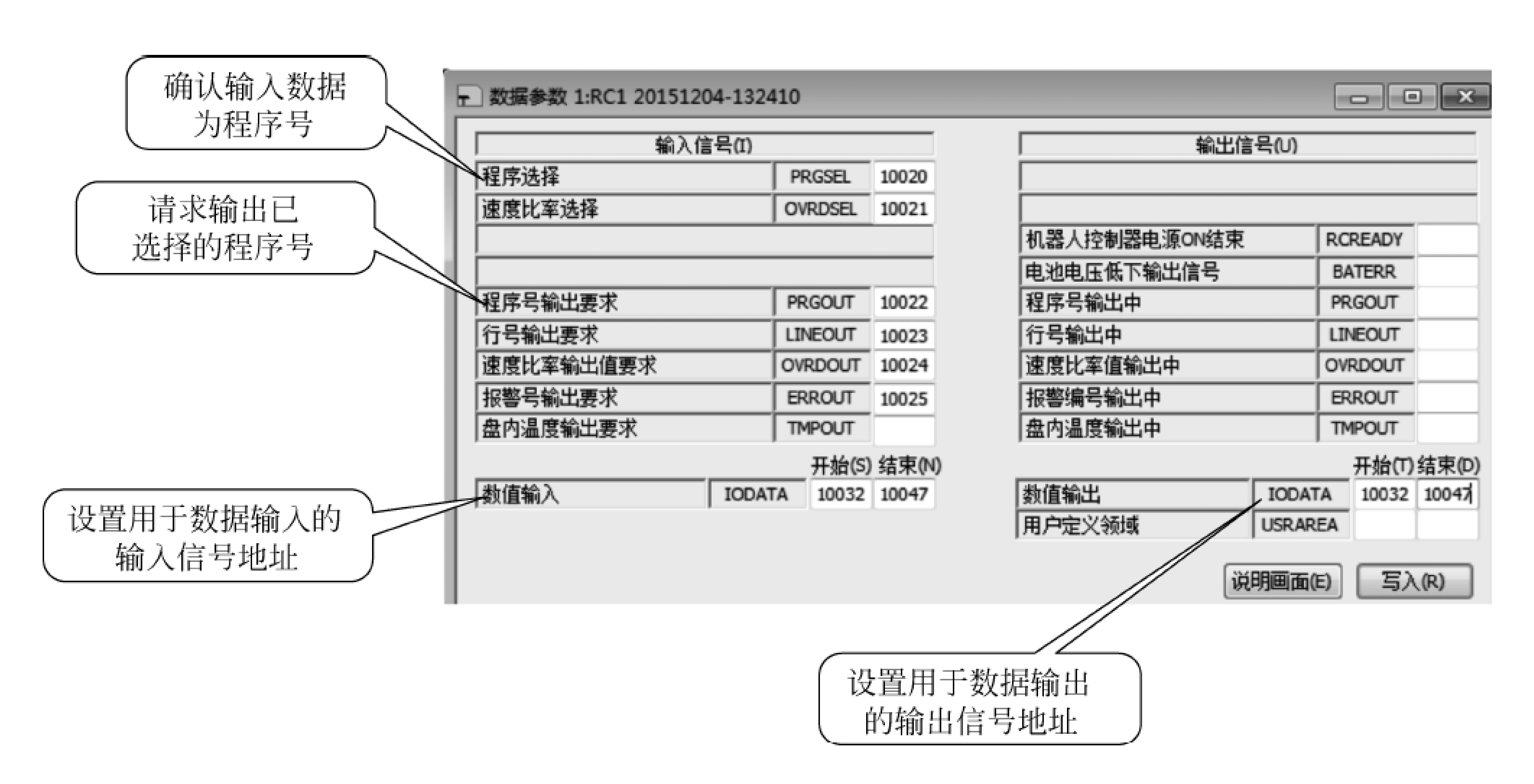


图 24-14　机器人软件一侧的设置

(3) 在机器人一侧,还必须定义输入数据的用途,输入的数据是作为程序号还是速度倍率,因此还有一个数据用途的确认键。参数“PRGSEL(程序选择)”就是将输入的数据确认为程序号。

24.4.2 程序号输出

当选择程序号完成后,必须在GOT上进行显示,以确保所选择程序号的正确性。

制作方法如下,参见图24-15。

(1) 在GOT上制作一个数据输出器件,该器件用于显示输出数据。器件的地址号根据表24-2设置为U3E1-10002。



图24-15　输出器件号的设置

(2) 在机器人参数中,IODATA参数用于设置数据输出的一组输出信号的起始地址。将与GOT器件U3E2-10002对应的输入地址10032～10047设置到参数IODATA中。

对机器人一侧,还必须定义输出数据的用途,输出数据是作为程序号还是速度倍率,因此还有请求输出数据的确认键。参数PRGOUT(请求输出程序号)就是将输出数据确认为程序号。

24.5 速度倍率的设置和显示

在GOT上设置和修改速度倍率也是客户最基本的要求。在GOT上设置和修改速度倍率的方法如下,如图24-16所示是在GOT上制作的速度倍率设置界面。

24.5.1 速度倍率的设置

(1) 在GOT上制作一个数据输入器件,该器件用于输入速度倍率。器件的地址号根据表24-1设置为U3E0-10002。

为使用方便,使用一键输入方法,即在GOT上制作10个按键。每个按键用于输入不同的速度倍率值。图24-17是“速度倍率＝60％”的按键制作。

(2) 在机器人参数中,IODATA参数是用于设置数据输入的一组输入信号的起始地址。将与GOT器件U3E0-10002对应的输入地址10032～10047设置到参数IODATA中,如图24-18所示。

这种设置方法可以设置任意的“速度倍率”。

(3) 在机器人一侧,还必须定义输入数据的用途,输入的数据是作为程序号还是速度倍

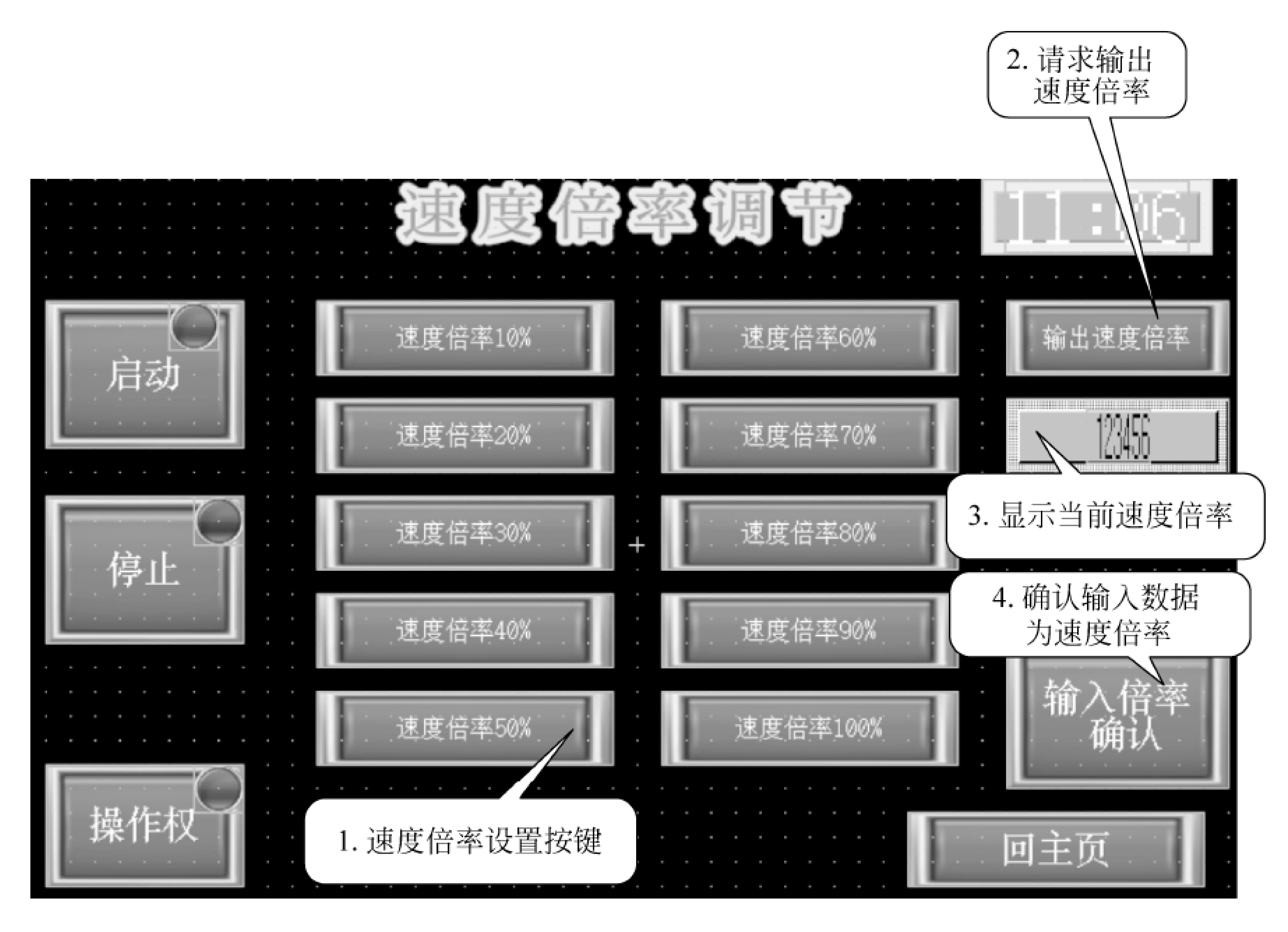


图 24-16 GOT 上制作的“速度倍率”设置界面

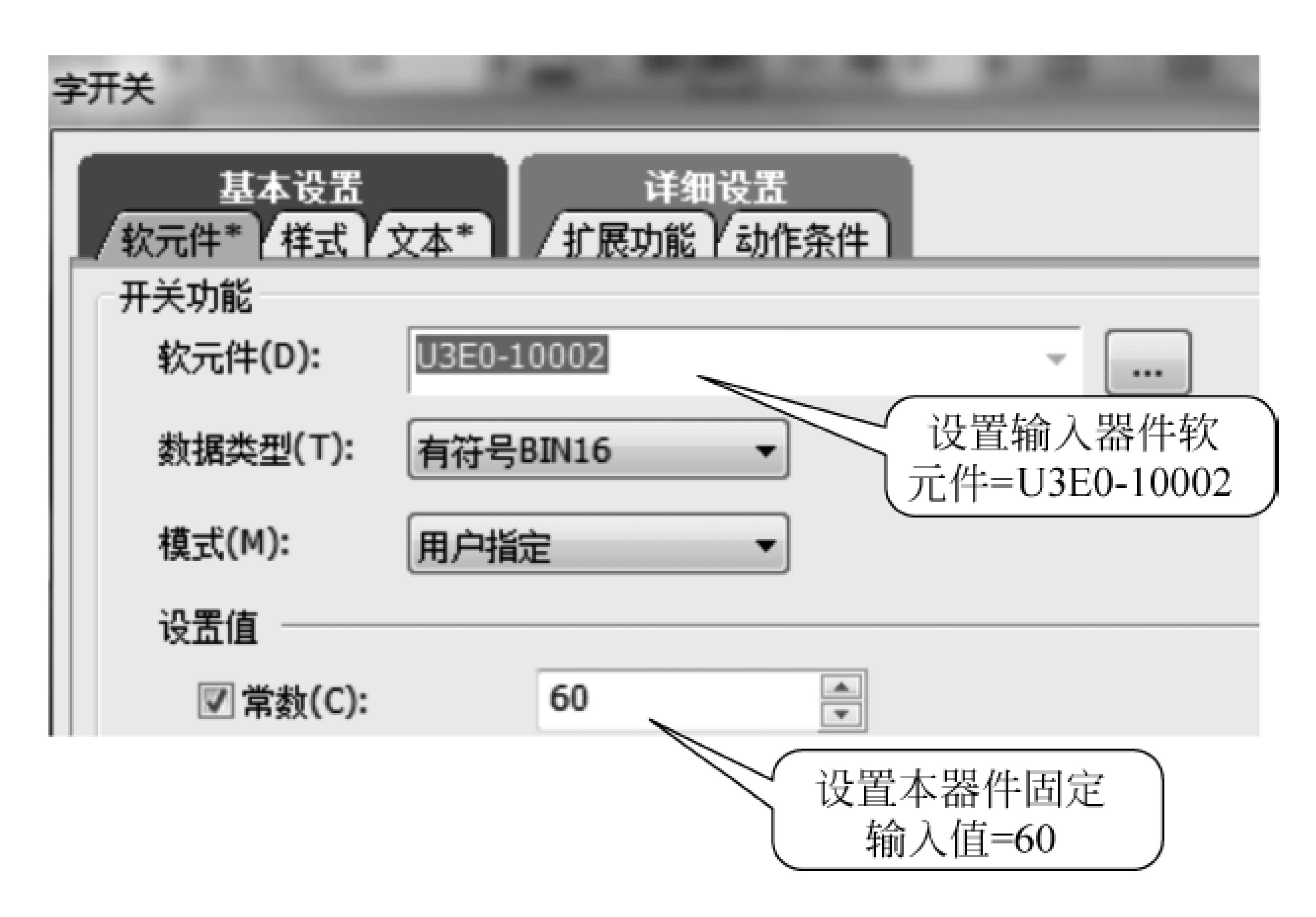


图 24-17 “速度倍率＝60％”的按键制作

率，因此还有一个数据用途的确认键。参数 OVRDSEL（速度倍率选择）就是将输入的数据确认为速度倍率。

24.5.2 速度倍率的输出

当选择速度倍率完成后，必须在 GOT 上进行显示，以确保所选择速度倍率的正确性。制作方法如下。

（1）在 GOT 上制作一个数据输出器件，该器件用于显示输出数据。器件的地址号根据表 24-2 设置为 U3E1-10002，如图 24-19 所示。

（2）在机器人参数中，IODATA 用于设置数据输出的一组输出信号的起始地址。将与



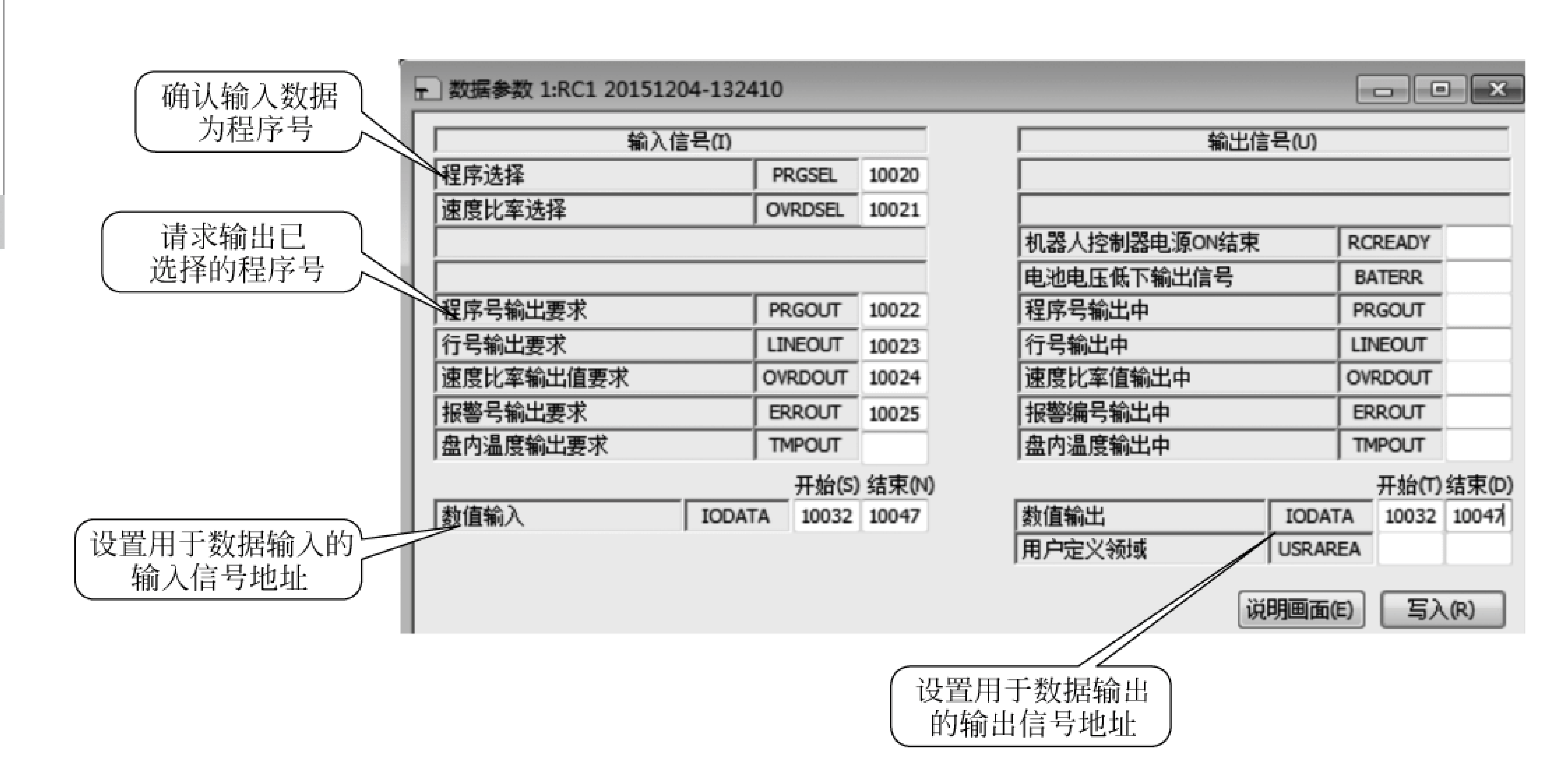


图 24-18　在机器人软件一侧的设置



图 24-19　“数据输出”器件的设置

GOT 器件 U3E2-10002 对应的输入地址 10032～10047 设置到参数 IODATA 中。

(3) 对机器人一侧，还必须定义输出数据的用途，输出数据是作为程序号还是速度倍率？因此还有请求输出数据的确认键。参数“OVRDOUT(请求输出‘速度倍率’)”就是将输出数据确认为速度倍率。

24.6　机器人工作状态读出及显示

如第 6 章所示，有大量表示机器人工作状态的变量。这些数字型变量也可以经过处理后在 GOT 上显示，图 24-20 就是各轴工作负载率(工作电流)的显示。其制作方法如下。

从表 24-2 可知，可以使用的输出信号范围为 10000～18191，这些输出信号对应在 GOT 上为字器件 U3E1-10000～10511。即有 512 个字器件可供使用。

只要在机器人程序中编制某一状态变量与输出信号的关系，就可以将状态变量显示在 GOT 上。

显示各轴工作负载率(工作电流)的制作方法如下。



图 24-20　各轴工作负载率显示屏

(1) 在 GOT 上使用 U3E1-10016～10021 作为 J1～J6 轴的工作电流显示部件，如表 24-3 所示。

表 24-3　输出信号对应表

序　　号	GOT	机器人(out)	功　　能
1	U3E1-10016	10256～10271	J1 轴工作电流
2	U3E1-10017	10272～10287	J2 轴工作电流
3	U3E1-10018	10288～10303	J3 轴工作电流
4	U3E1-10019	10304～10319	J4 轴工作电流
5	U3E1-10020	10320～10335	J5 轴工作电流
6	U3E1-10021	10336～10351	J6 轴工作电流

(2) 在机器人程序中编制一个子程序，专门用于显示工作电流，在需要时调用该程序。

程序号 60：

```
M_OUTw(10256) = M_LdFact(1)'第 1 轴工作电流
M_OUTw(10272) = M_LdFact(2)'第 2 轴工作电流
M_OUTw(10288) = M_LdFact(3)'第 3 轴工作电流
M_OUTw(10304) = M_LdFact(4)'第 4 轴工作电流
M_OUTw(10320) = M_LdFact(5)'第 5 轴工作电流
M_OUTw(10336) = M_LdFact(6)'第 6 轴工作电流
```

运行该程序后就在 GOT 上显示了各轴的工作电流。

24.7　JOG 界面制作

JOG 是机器人的一种工作模式，在没有示教单元的场合，可以使用 GOT 进行 JOG 操作。

制作 JOG 界面，关键在于机器人一侧的参数设置。有关 JOG 操作的输入信号地址是固定的，不可随意设置，如果随意设置，机器人系统会报警。必须按表 24-4 和图 24-21 设置

分配给JOG的输入输出信号。

表 24-4　输入信号对应表

序　　号	GOT	机器人（被固定分配）	规定对应的功能信号	
			功能	参数名称
1	U3E0-10005.b12	10092	JOG有效	JOGENA
2	U3E0-10006.b0	10096	J1+	
3	U3E0-10006.b1	10097	J2+	
4	U3E0-10006.b2	10098	J3+	
5	U3E0-10006.b3	10099	J4+	
6	U3E0-10006.b4	10100	J5+	
7	U3E0-10006.b5	10101	J6+	
8	U3E0-10006.b6	10102	J7+	
9	U3E0-10006.b7	10103	J8+	
10	U3E0-10007.b0	10112	J1－	
11	U3E0-10007.b1	10113	J2－	
12	U3E0-10007.b2	10114	J3－	
13	U3E0-10007.b3	10115	J4－	
14	U3E0-10007.b4	10116	J5－	
15	U3E0-10007.b5	10117	J6－	
16	U3E0-10007.b6	10117	J7－	
17	U3E0-10007.b7	10117	J8－	
18	U3E0-10005.b13	10093	JOG直交模式	
19	U3E0-10005.b14	10094	JOG关节模式	
20	U3E0-10005.b15	10095	JOG Tool模式	



图 24-21　分配给JOG的输入输出信号

由于在机器人一侧已经规定了输入输出信号地址，所以在GOT一侧的器件号也就被规定了。

根据表24-4制定GOT画面中的各按键即可，如图24-22所示。



图 24-22 JOG 操作屏的制作与设置

24.8 需要思考的问题

(1) GOT 与机器人连接时通信参数如何设置?

(2) GOT 中的软元件如何与机器人中的输入输出信号相对应?

(3) 如何制作选择程序号和显示程序号的 GOT 画面?

(4) 如何制作设置速度倍率和显示速度倍率的 GOT 画面?

(5) 如何制作显示机器人工作电流的 GOT 画面?

(6) 如何制作显示机器人工作状态变量的 GOT 画面?

第 25 章 第 25 日——机器人在抛光研磨项目中的应用

【学习目的】

在工件抛光项目上经常使用机器人。本章要学习根据工作路径编制程序的方法。抛光项目中关键的技术是检测工件与抛光轮之间的压力，本章提供了一种解决方案。在抛光项目中，还有许多与抛光轮速度、材质、磨料相关的工艺参数，需要在实践中摸索。

25.1 项 目 综 述

某客户的工件要求采用机器人抓取实现抛光。工件如图 25-1 所示，要求抛光 5 个面。抛光运行轨迹由机器人完成。

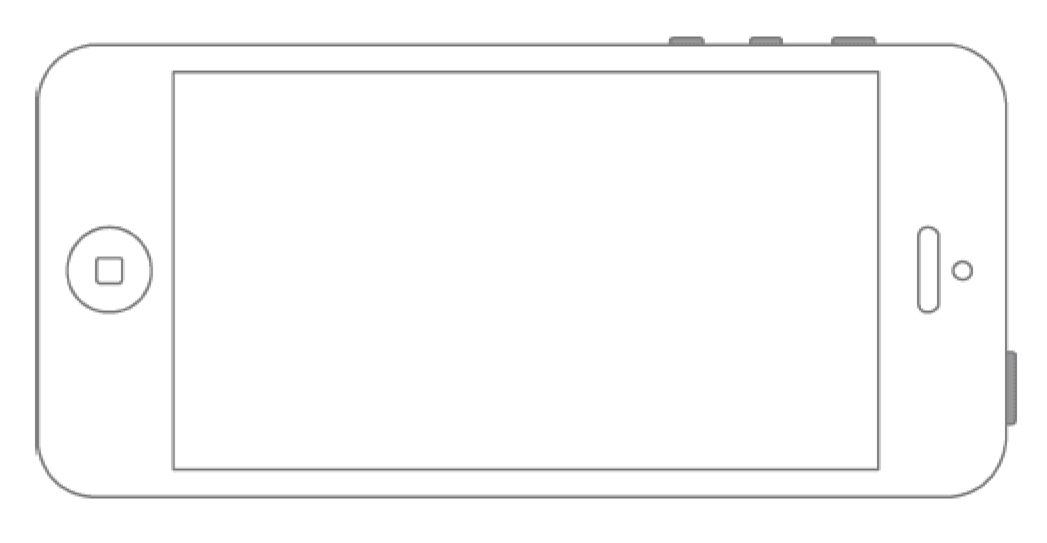


图 25-1　加工工件

(1) 抛光轮由变频电机驱动，必须能够预置多种速度。

(2) 工件由机器人夹持实施抛光，抛光面 5 个。

(3) 机器人由两套系统控制——外部硬件操作屏与触摸屏 GOT 构成。在触摸屏上可以设置各种工作参数，例如“位置补偿值”。

(4) 能够简单检测抛光质量。

(5) 要求机器人运行轨迹符合工件的 3D 轨迹。

(6) 能够进行工件计数。

(7) 夹持工件不需要视觉装置辅助调整。

(8) 能够提供实用的工艺参数(抛光轮速度，机器人运行线速度，抛光轮及磨料)。

(9) 抛光精度<0.12mm。

(10) 成本低。

25.2 解 决 方 案

25.2.1 硬件配置

1. 硬件配置一览表

经过技术经济分析，决定采用如表 25-1 所示硬件配置。

表 25-1　硬件配置

序　　号	名　　称	型　　号	数　　量	备　　注
1	机器人	RV-2F	1	三菱
2	简易示教单元	R33TB	1	三菱
3	输入输出卡	2D-TZ368	1	三菱
4	PLC	FX3U-32MR	1	三菱
5	触摸屏	GS1000	1	三菱
6	变频器	A740-2.2K	1	三菱
7	电机	普通电机 2kW	1	
8	光电开关		1	

2. 硬件配置说明

硬件选配以三菱机器人 RV-2F 为中心，该机器人为 6 轴机器人，由于需要对 5 个工作面进行抛光作业，所以必须选择 6 轴机器人，同时工件加抓手重量小于 2kg，所以选用搬运重量为 2kg 的机器人。

(1) 机器人选用 RV-2F，其工作参数主要指标如下。

① 夹持重量：2kg。

② 臂长：504mm。

③ 标配控制器：CR751D。

(2) 示教单元：R33TB(必须选配，用于示教位置点)。

(3) 机器人选件：输入输出信号卡 2D-TZ368(32 输入/32 输出)，用于接收外部操作屏信号和控制外围设备动作。

(4) 选用变频电机＋三菱变频器 A740-2.2K 作为抛光轮驱动系统。速度可调。

(5) 选用三菱 PLC FX3U-32MR 作为主控系统。

(6) 触摸屏选用 GS1000 系列。触摸屏可以直接与机器人相连接，直接设置和修改各工艺参数。

25.2.2　应对客户要求的解决方案

1. 解决方案

(1) 抛光轮由变频器＋普通电机驱动，由 PLC 控制可以预置 7 种速度；速度值可以修改。

(2) 工件由机器人夹持实施抛光。机器人搬运重量为 2kg，6 轴，臂长 504mm。可实现复杂的空间运行轨迹。

(3) 触摸屏 GS1000 可以直接与机器人相连接，直接设置和修改各工艺参数。

(4) 使用机器人的负载检测控制，间接实现抛光质量检测。

(5) 在进料输送端设置挡块，使工件定位。抓手为内张型抓手，可以控制定位位置。同时放大抛光行程，可以满足工件表面全抛光的要求。

(6) 工件计数由卸料端光电开关检测，有 PLC 计数，在 GOT 上显示。

(7) 机器人重复定位精度为 0.02mm，可以满足<0.12mm 的要求。

(8) 抛光工艺参数必须通过工艺实验确定。以下是工艺实验方案。

2. 工艺实验方案

1）抛光轮材料、磨料、速度与抛光工件质量的关系

在机器人运行速度确定和抛光磨料确定的条件下测试抛光轮材料、速度与工件抛光质量的关系。

实验时以不同的磨轮（不同材料的磨轮）在不同的转速下做实验。实验记录表格如表 25-2～表 25-4 所示。

表 25-2　磨轮速度与抛光工件质量的关系 1

机器人运行速度（秒/件）　　抛光磨料：甲

	速度 1	速度 2	速度 3	速度 4	速度 5
抛光轮 A					
抛光轮 B					
抛光轮 C					
抛光轮 D					
抛光轮 E					

表 25-3　磨轮速度与抛光工件质量的关系 2

机器人运行速度（秒/件）　　抛光磨料：乙

	速度 1	速度 2	速度 3	速度 4	速度 5
抛光轮 A					
抛光轮 B					
抛光轮 C					
抛光轮 D					
抛光轮 E					

表 25-4　磨轮速度与抛光工件质量的关系 3

机器人运行速度（秒/件）　　抛光磨料：丙

	速度 1	速度 2	速度 3	速度 4	速度 5
抛光轮 A					
抛光轮 B					
抛光轮 C					
抛光轮 D					
抛光轮 E					

2）工件抛光质量与工作电流的关系

必须测定最佳工作电流。因为磨轮是柔性磨轮，无法预先确定运行轨迹，而工作电流表示了工件与磨轮的贴合程度（磨削量），所以必须在基本选定磨轮转速和工件运行速度后，测定最佳工作电流。只有达到最佳工作电流，才能被认为是正常抛光完成。实验时需要逐步加大抛光磨削量以观察工作电流的变化，要注意磨削量在图纸给出的加工范围内。

工件抛光质量与工作电流的关系如表 25-5 所示。

表 25-5　工件抛光质量与工作电流的关系

序　　号	工 作 电 流	工件抛光质量
1		
2		
3		
4		
5		
6		

25.3　机器人工作程序编制及要求

编制程序的要求如下。

(1) 必须按工件的 3D 轮廓编制运行轨迹。不采用描点法。

(2) 能够设置 1 次、2 次、3 次磨削量。能够设置磨轮转速。

(3) 能够根据磨轮材料自动匹配磨轮转速、工件线速度。根据每一工件的最少加工时间(效率)确定工件运动线速度。

(4) 自动添加抛光磨料。

(5) 有工件计数功能。

25.3.1　工作流程图

如图 25-2 所示为抛光工件总流程图。主要核心在于有一个试磨程序——即通过检测工作负载率测试工件与抛光轮的贴合紧密程度，如果达到最佳工作电流就进入正常抛光工作程序，如果未达到最佳工作电流就进入基准工作点补偿程序，所以最佳工作电流是工件抛光质量的间接反映。

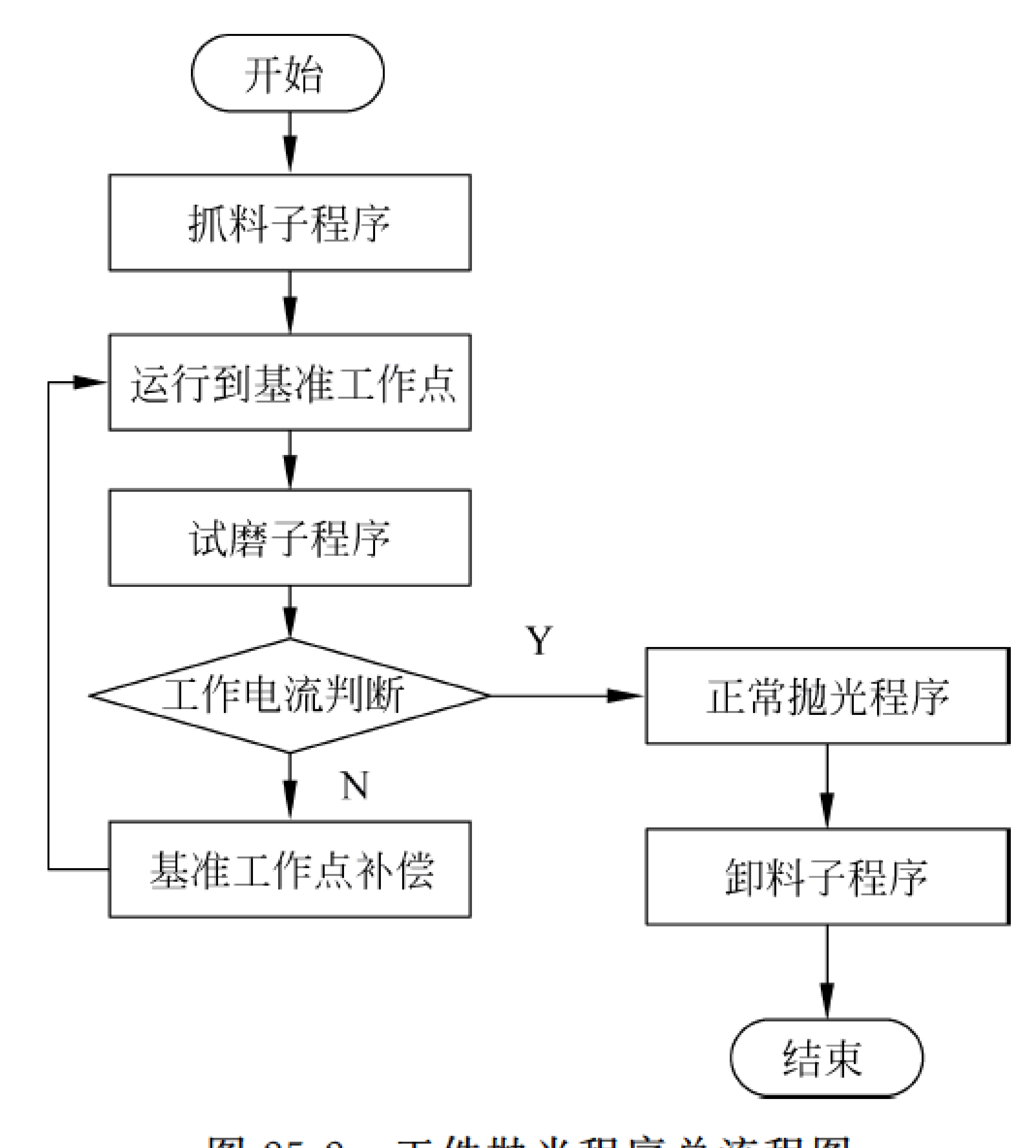


图 25-2　工件抛光程序总流程图

正常抛光工作流程如图 25-3 所示，包括背面抛光和其余 4 个面的圆弧抛光。

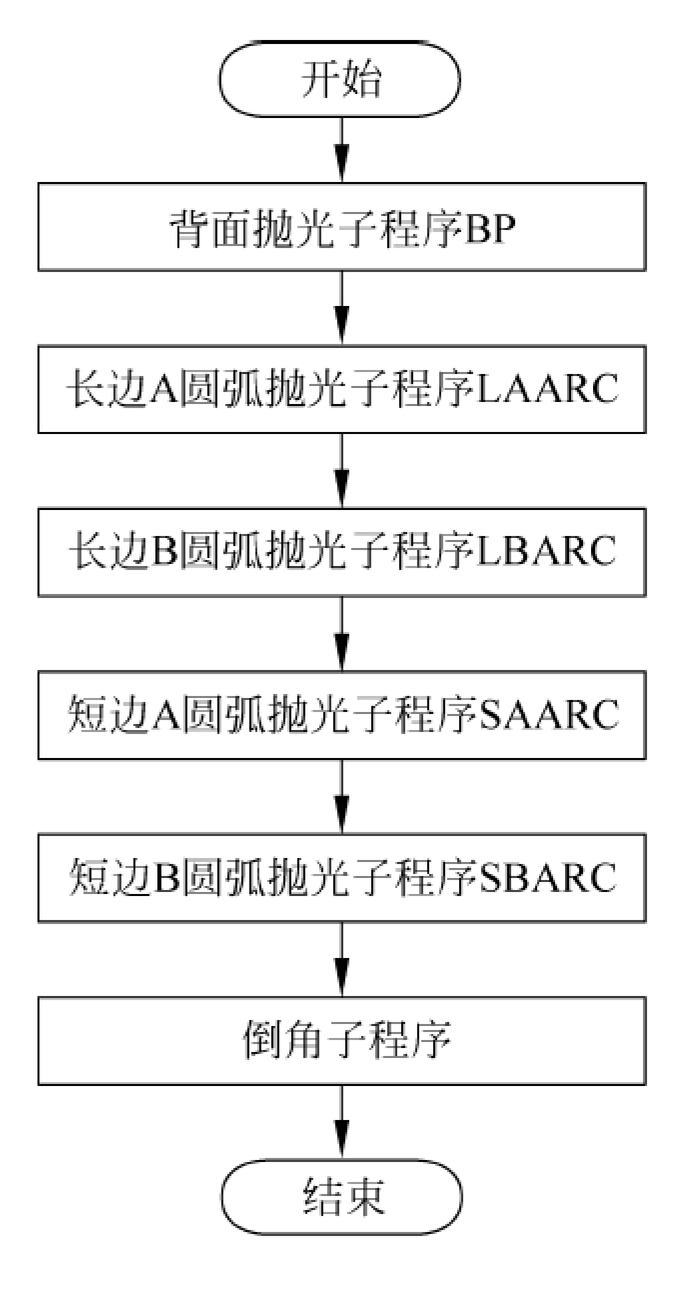


图 25-3　正常抛光工作流程

25.3.2　子程序汇总表

由于 5 个面的抛光运行轨迹各不相同，为简化编程，预先将各部分动作划分为若干子程序。经过分析，需要编制的子程序如表 25-6 所示。

表 25-6　子程序汇总

序　　号	子程序名称	功　　能	程 序 代 号
1	初始化程序	进行初始化	CSH
2	抓料子程序	抓料	ZL
3	试磨及电流判断子程序	试磨/电流判断/基准点补偿	ACTEST
4	背面抛光子程序	抛光背面	BP
5	长边 A 抛光子程序	抛光长边 A 圆弧	LAARC
6	长边 B 抛光子程序	抛光长边 B 圆弧	LBARC
7	短边 A 抛光子程序	抛光短边 A 圆弧	SAARC
8	短边 B 抛光子程序	抛光短边 B 圆弧	SBARC
9	圆弧抛光子程序	纯粹圆弧抛光	YHPG
10	圆角抛光子程序 1	抛光圆角	ARC1
11	圆角抛光子程序 2	抛光圆角	ARC2
12	圆角抛光子程序 3	抛光圆角	ARC3
13	圆角抛光子程序 4	抛光圆角	ARC4
14	卸料子程序	卸料	XIEL

25.3.3 抛光主程序

为编程简洁明了，将工作程序分解为若干子程序，主程序则负责调用这些子程序。抛光主程序 MAIN100：

```
CALLP  "CSH"'——调用初始化子程序。
 CALLP "ZL"'——调用抓料子程序。
 CALLP "ACTEST"'——调用试磨电流判断子程序。
  * ZC'——正常抛光程序运行标记。
 CALLP "BP"'——调用背面磨子程序。
 CALLP "LAARC"'——调用长边 A 抛光子程序。
 CALLP "LBARC"'——调用长边 B 抛光子程序。
 CALLP "SAARC"'——调用短边 A 抛光子程序。
 CALLP "SBARC"'——调用短边 B 抛光子程序。
 CALLP "ARC1"'——调用圆弧倒角子程序 1。
 CALLP "ARC2"'——调用圆弧倒角子程序 2。
 CALLP "ARC3"'——调用圆弧倒角子程序 3
 CALLP "ARC4"'——调用圆弧倒角子程序 4
 CALLP "XIEL"'——调用卸料子程序。
 END'——主程序结束。
```

25.3.4 初始化子程序

初始化程序用于对机器人系统的自检和外围设备的启动和检测。初始化程序如下。

初始化程序 CSH：

```
 '初始化程序
 *  CSH'——初始化程序标签
M_OUT(10) = 1'——抛光轮启动。
DLY 0.5'——暂停。
M_OUT(11) = 1'——气泵启动。
M10 = M_IN(10)'——M_IN(10)气压检测。
M11 = M_IN(11)'——M_IN(11)抛光轮速度到位检测。
M12 = M_IN(12)'——M_IN(12)输送带有料无料检测。
 M15 = M10 + M11 + M12
'——判断气压,抛光轮速度,有料信号是否全部到位。
 IF M15 = 3 THEN'——判断。
GOTO  *  ZL'——跳转到抓料子程序。
 ELSE'——否则
GOTO  *  CSH'——跳转回到初始化子程序。
 endif'——选择语句结束。
END'——主程序结束。
* ZL'——抓料子程序标签。
'——如果气压、抛光轮速度、有料信号全部到位,就进入抓料程序,否则继续进行初始化程序。
```

25.3.5 电流判断子程序

```
* ACTEST'电流检测程序
```

```
M52 = M_LdFact(2)'——检测 2 轴负载率。
M53 = M_LdFact(3)'——检测 3 轴负载率。
M55 = M_LdFact(5)'——检测 5 轴负载率。
M60 = M52 + M53 + M55'——M60 为综合负载率。
P1 = P1 + P101'——P101 为磨削补偿量。
 Mov P1'——P1 试磨起点(基准点)。
MVS P2'——试磨终点。
 If M60 < M_100 Then'——"工作电流判断"(M_100 为工艺规定数据,可以设定)。如果综合负载
率小于工艺规定数据,则
P101.X = P101.X + 0.01'—— P101 为磨削补偿(对试磨基准点进行补偿)。
GOTO * ACTEST'——重新试磨。
Else'——否则
GOTO * PG100'—— PG100 为正常抛光程序。
EndIf'——选择语句结束。
END'——主程序结束。
```

25.3.6 背面抛光子程序

1. 背面抛光运行轨迹

背面抛光程序必须考虑做三次抛光运行。每一次比前一次有一个微前进量。背面抛光运行轨迹如图 25-4 所示。以 P1 点为基准点,其余 P2、P3、P4 各点根据 P1 点计算。运行轨迹为 P1→P2→P3→P4→P5→P6→P3→P4→P7→P8→P3。

背面抛光运行轨迹

图 25-4 背面抛光运行轨迹

2. 背面抛光子程序 BP

```
P2 = P1 - P_10'——P_10 为工件长。
P3 = P2 - P_11'——P_11 为退刀量。
P4 = P3 + P_10'——赋值。
'——第 1 次粗抛光循环。
MOV P1'——P1 为测定的基准点。
MVS P2'——移动到 P2 点。
MVS P3'—— 移动到 P3 点。
MVS P4'——移动到 P4 点。
'——第 2 次抛光循环
MVS P1 + P101'—— P101 为 1#进刀量。
MVS P2 + P101'——移动到"P2 + P101"。
MVS P3'——移动到"P3 点"。
MVS P4'——移动到"P4 点"。
'——第 3 次抛光循环
MVS P1 + P102'——P102 为 2#进刀量。
MVS P2 + P102'——移动到"P2 + P102"。
MVS P3'——移动到"P3 点"。
MVS P4'——移动到"P4 点"。
END'——主程序结束。
```

25.3.7 长边 A 抛光子程序

1. 长边圆弧的抛光运行轨迹

长边圆弧的抛光运行轨迹分为上半圆弧运行轨迹和下半圆弧运行轨迹,如图 25-5 所

示。这是因为抛光轮的抛光工作线是一直线，而且是按一个方向旋转(图中是顺时针方向)，为简化编程，将其分为上半圆弧运行轨迹和下半圆弧运行轨迹。

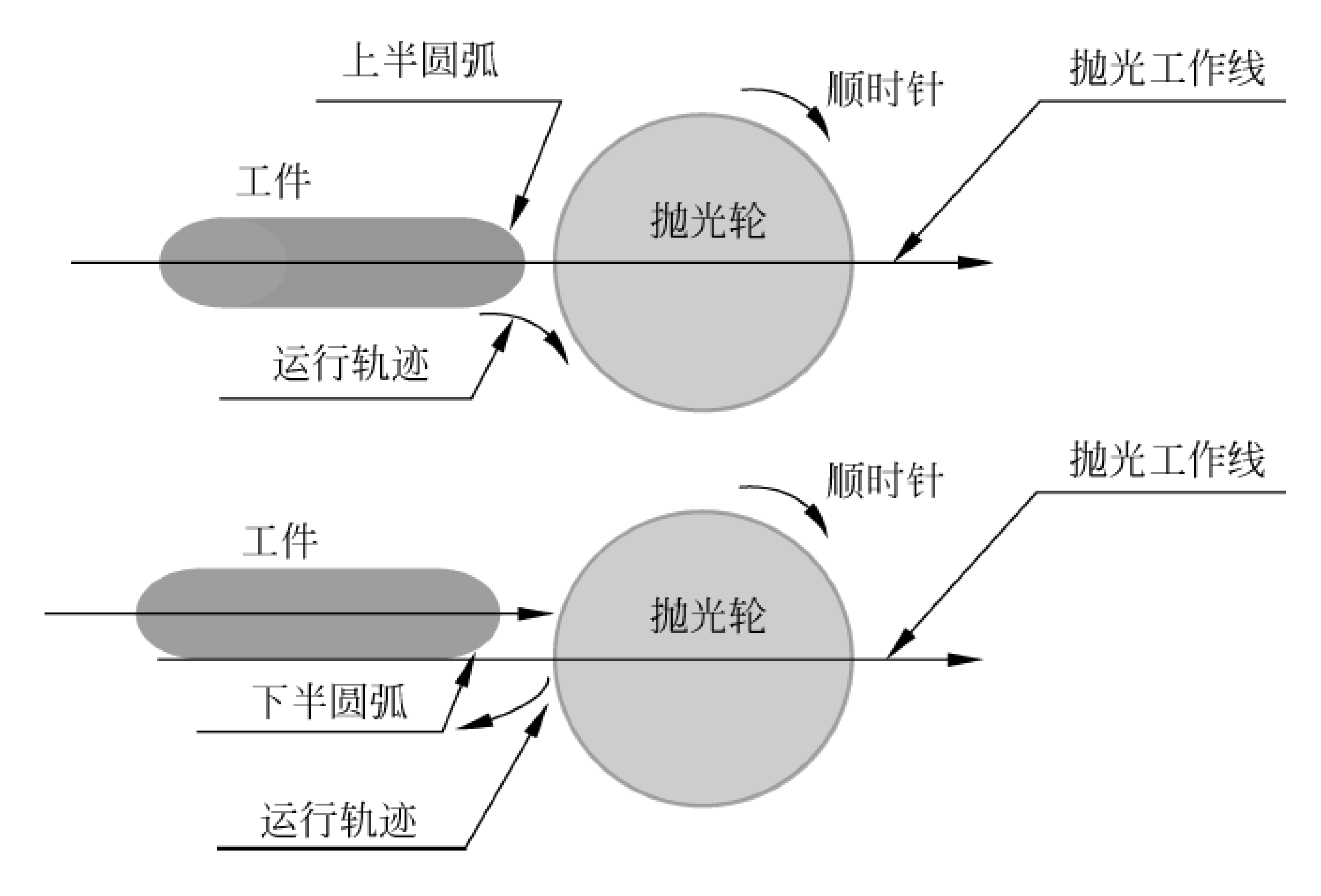


图 25-5　长边圆弧抛光运行轨迹示意图

2. 上半圆弧运行程序 YHARC

1）上半圆弧运行轨迹

上半圆弧运行轨迹如图 25-6 所示。

2）上半圆弧运行程序 YHARC

```
Ovrd 20'——设置速度倍率。
P10 = P_Curr'——取当前点为 P10。
P11 = P_Curr - P_38'—— P_38 是圆弧插补终点数据。P11 为圆弧插补终点。
P12 = P_Curr - P_36'——P_36 是圆弧插补半径数据。P12 为圆心。
MVR3 P10,P11,P12'——圆弧插补,抛光运行。
MVR3 P11,P10,P12'——圆弧插补,回程。
MVR3 P10,P11,P12'——圆弧插补,抛光运行。
MVR3 P11,P10,P12'——圆弧插补,回程。
MVR3 P10,P11,P12'——圆弧插补,抛光运行。
MVR3 P11,P10,P12'——圆弧插补,回程。
End'——主程序结束。
```

3. 下半圆弧运行程序

1）下半圆弧运行轨迹

下半圆弧运行轨迹如图 25-7 所示。

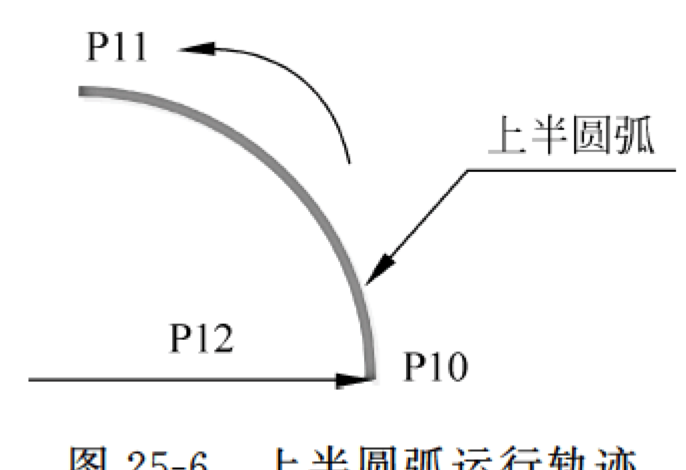


图 25-6　上半圆弧运行轨迹

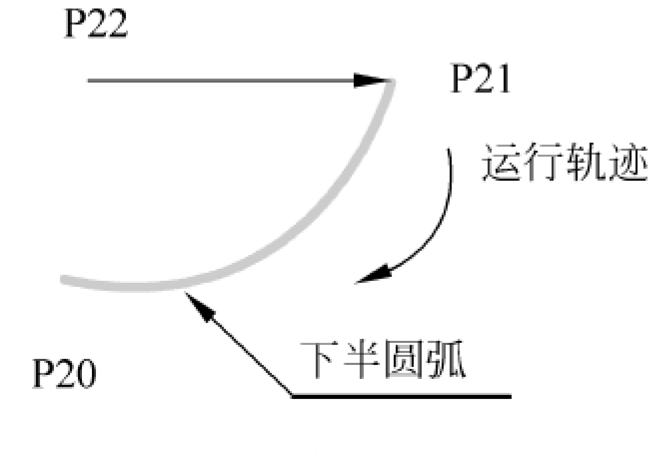


图 25-7　下半圆弧运行轨迹

2）下半圆弧运行程序

```
Ovrd 20'——设置速度倍率。
1  P20 = P_Curr'——取当前点为 P20。
2  P21 = P_Curr + P_38'——P_38 是圆弧插补终点数据。P11 为圆弧插补终点。
3  P22 = P_Curr + P_37'——P_37 是圆弧插补半径数据。P22 为圆心。
4  MVR3 P20,P21,P22'——圆弧插补,抛光运行。
5  MVR3 P21,P20,P22'——圆弧插补,回程。
6  MVR3 P20,P21,P22'——圆弧插补,抛光运行。
7  MVR3 P21,P20,P22'——圆弧插补,回程。
8  MVR3 P20,P21,P22'——圆弧插补,抛光运行。
9  MVR3 P21,P20,P22'——圆弧插补,回程。
10 End'——主程序结束。
```

25.3.8 圆弧倒角子程序 1

1. 圆弧抛光的运行轨迹

本工件有 4 个圆弧,圆弧抛光的运行轨迹如图 25-8 所示。

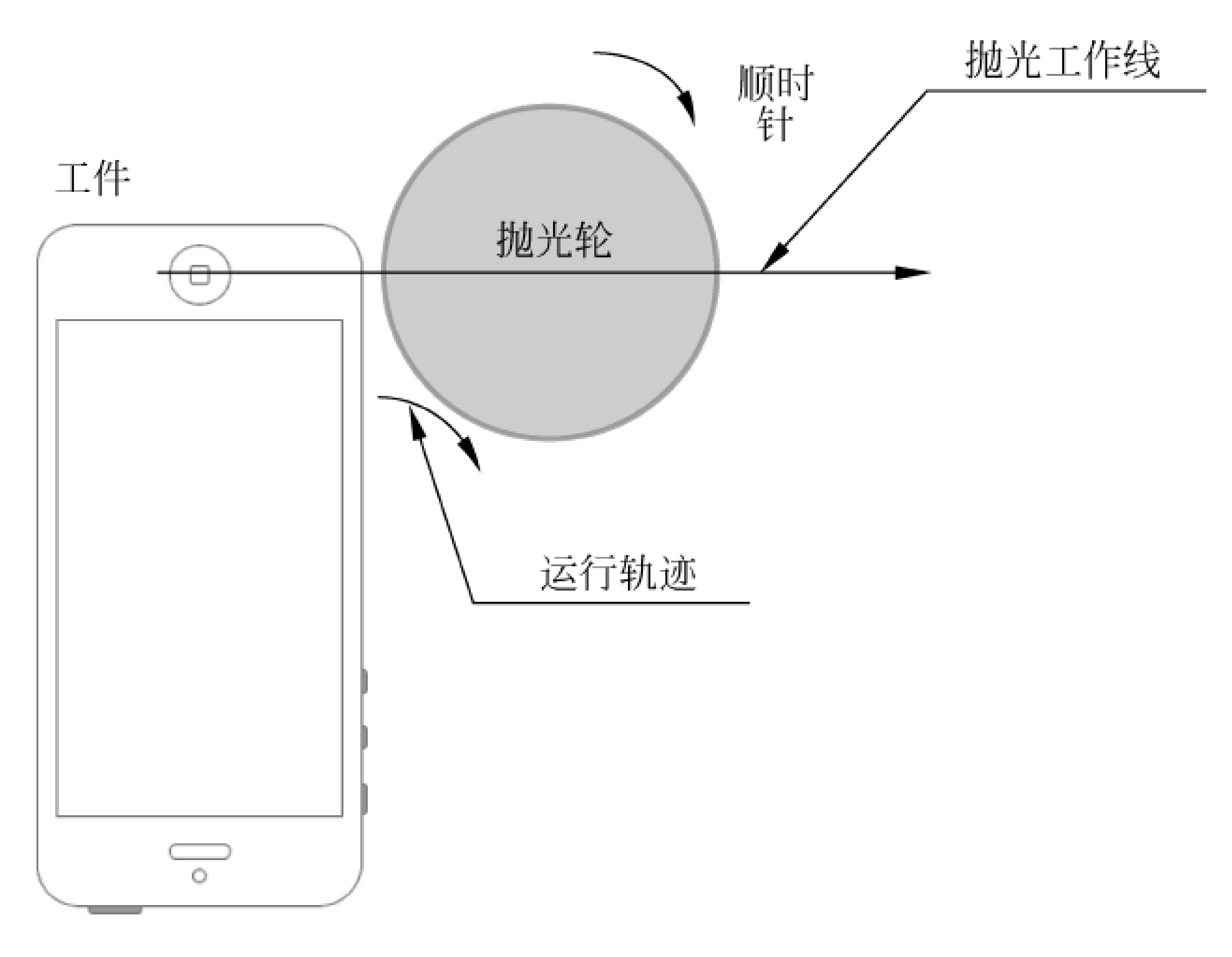


图 25-8 圆弧抛光的运行轨迹

圆弧倒角子程序用于对工件的 4 个圆角进行抛光。由于机器人的控制点设置在工件中心,所以可以直接将工件运行到如图位置后,进行圆弧插补。

2. 圆弧倒角子程序 ARC1

```
1 Ovrd 20'——设置速度倍率。
2 P10 = P_Curr'——P10 为当前位置点。
3 P11 = P_Curr + P_28'——P_28 是圆弧终点数据。
4 P12 = P_Curr - P_26'——P_26 是圆弧半径。P12 是圆心。
5 Mvr3 P10,P11,P12'——圆弧倒角,抛光。
6 Mvr3 P11,P10,P12'——圆弧倒角,回程。
7 Mvr3 P10,P11,P12'——圆弧倒角,抛光。
8 Mvr3 P11,P10,P12'——圆弧倒角,回程。
```

```
9  Mvr3 P10,P11,P12'——圆弧倒角,抛光。
10 Mvr3 P11,P10,P12'——圆弧倒角,回程。
11 Mvr3 P10,P11,P12'——圆弧倒角,抛光。
12 End'——主程序结束。
```

“圆弧倒角子程序”与“长边圆弧抛光程序”在结构上是相同的,都是对工件运行圆弧轨迹,只是各自的圆弧起点、终点、圆弧半径圆心位置各不相同,需要做不同的设置。

25.3.9 空间过渡子程序

1. 概说

工件为矩形,由于工件有 5 个面需要抛光,抛光磨削工作线为抛光轮直径水平线,其位置是固定的。在机器人坐标系中,其 X、Z 坐标是固定的,Y 坐标取抛光轮中心线。因此,编程序的工作是使工件待抛光面与抛光轮磨削工作线有相对运动。

由于机器人夹持工件,为编程方便,设置“机器人控制点”为工件矩形背面中心点(计入了抓手长度因素)。

基准抛光点为背面磨削的起点,即工件底部中心点位于磨削线 Z 向 10mm 处。该点用全局变量 P_01 表示,即在各全部程序中有效。

为了将各待抛光面移动到抛光轮工作线,需要进行空间移动。机器人的“形位”会改变。其中,绕 X/Y/Z 轴的旋转是通过两个点的乘法进行的。本节是编制空间过渡点程序。通过该程序实现了各子程序的连接。图 25-9 是工件尺寸示意图。

2D
F
E=F–L
2L

图 25-9 工件尺寸示意图

2. 专用工作位置点

为编程需要,必须预置专用工作点。专用工作位置点如表 25-7 所示。

表 25-7 专用工作位置点

序 号	变量名称	变量内容	变量类型
1	P_01	抛光基准点	全局
2	P_02	L——工件 1/2 长度	全局
3	P_03	D——工件 1/2 宽度	全局
4	P_04	H——工件 1/2 厚度	全局
5	P_05	E——工件数据(斜边-直边)	全局
6	P_06	(0,0,0,0,90,0)	绕 Y 轴旋转 90° 全局
7	J_6	(0,0,0,0, 0,90)	J6 轴旋转 90° 全局

3. 抛光主程序 MAIN100 的核心部分

```
(CALLP "BP"调用背面抛光子程序)
1  MOV P_01'——回到基准点。
```

2　MVS P_01 - (2L,0,0,0,0,0)'——X 方向退 2L(L = 1/2 工件长度),D = 1/2 工件宽度,E = 斜边减长边(取 E = 20)。

4　MOV P_CURR * P_06'——B 轴旋转 90°(成水平面)。

5　MOV P_CURR - (0,0,L + 10,0,0,0)'——Z 方向下行 L + 10mm。

6　MOV P_CURR + (L + R,0,0,0,0,0)'——X 方向前进 L + R,到圆弧插补起点。

7　CALLP "SAARC"'——做圆弧插补(三次)抛光短边 1。

8　MOV P_CURR - (E,0,0,0,0,0)'——X 方向退 E(斜边 - 长边),准备磨长边 1。

9　MOV J_CURR + (0,0,0,0,0,90)'——Z 轴旋转 90°。

10　MOV P_CURR + (X1,0,0,0,0,0)'——X 方向前进(L - D + E) + R,到圆弧插补起点(X1 = L - D + E + R)。

11　CALLP "LAARC"'——做圆弧插补(三次)磨长边 1。

12　MOV P_CURR - (X1,0,0,0,0,0)'——X 方向退(L - D + E) + R,准备磨短边 2。

13　MOV J_CURR + (0,0,0,0,0,90)'——Z 轴旋转 90°。

14　MOV P_CURR + (X2,0,0,0,0,0)'——X 方向前进(E + R),到圆弧插补起点(X1 = E + R)。

15　CALLP "SAARC"'——磨短边 2 做圆弧插补(三次)。

16　MOV P_CURR - (E,0,0,0,0,0)'——X 方向退(E),准备磨长边 2。

17　MOV J_CURR + (0,0,0,0,0,90)'——Z 轴旋转 90°。

18　MOV P_CURR + (X1,0,0,0,0,0)'——X 方向前进(L - D + E) + R,到圆弧插补起点。

19　CALLP "LAARC"'——做圆弧插补(三次)磨长边 2。

20　END'——主程序结束。

4. 关于运行轨迹的问题

(1) 设置 TOOL 坐标系时,要尽量减小对 Z 轴的设置,因为 Z 方向过大,则会出现如果需要摆角 90°时,机器人不能够完成的情况,所以在设计抓手时,应该尽量缩短抓手的长度。

一般地,机器人的位置控制点设置在抓手中心点(出厂设置在机械 IF 法兰中心点)。为了使抓手绕 XYZ 轴都能够旋转(而且能够旋转较大的角度),就必须设置 Z 坐标尽量小。

(2) 工件需要旋转某一角度时,使用点与点的乘法指令效果较好。使用点与点的加法有时可以得到同样的效果,有时得到意想不到的轨迹。

(3) 对于 TOOL 坐标系,可以绕其中某一点旋转,但运动轨迹不一定是需要的轨迹。

(4) 如果确定是直线运动,就必须用 MVS 指令。用 MOV 指令可能出现意想不到的轨迹。

(5) 要获得确切的轨迹,必须使用圆弧插补指令和直线指令。

(6) 尽量少使用全局变量,以免全局变量的改变影响所有程序。

25.4 结　束　语

抛光项目涉及的工艺因素很多,是比较复杂的应用类型。

(1) 从机器人的使用角度来考虑,主要是磨削工作电流的影响,因此在不同的磨轮材料和速度下,检测获得适当的工作负载电流值极其重要。

(2) 机器人的工作运行轨迹有多种编程方法,本文介绍的只是其中一种方法。

(3) 注意在圆弧磨削时的上半圆弧与下半圆弧的区别。

25.5 需要思考的问题

（1）如何规划抛光主程序？抛光主程序的程序结构包含哪些内容？

（2）如何编制机器人工作电流检测程序？

（3）如何编制基准工作点补偿程序？

（4）如何编制圆弧抛光子程序？

（5）如何编制空间过渡子程序？

第 26 章　第 26 日——机器人与视觉系统

【学习目的】

机器人与视觉系统的联合使用是扩展机器人应用范围的重要手段,目前已经得到广泛使用。本章要学习机器人与视觉系统联合使用的方法,重点是如何将视觉系统获得的信息转换为机器人系统的位置坐标。

26.1　概　　述

在很多流水线上,工件的摆放位置是任意的,当工件随流水线运行到机器人工作范围时,要求机器人能够识别这些工件的位置进行抓取。而实际上,机器人只能够根据机器人的坐标系动作,所以必须采用视觉系统——照相机通过照相——所获得的工件位置信息传送给机器人。视觉系统所获得的工件位置信息称为像素坐标,即根据视觉系统的坐标系所获得的坐标值。将视觉坐标系与机器人坐标系建立关系的工作称为标定。经过标定后,视觉系统传过来的位置信息可以为机器人所用。

本章介绍视觉系统在机器人上的应用。

26.2　前期准备及通信设置

26.2.1　基本设备配置及连接

1. 基本配置

为了进行视觉通信,最基本的硬件配置如表 26-1 所示。

表 26-1　视觉通信基本硬件配置

序　　号	设　　备	数　　量	说　　明
1	机器人	1	
2	视觉传感器	1	
3	相机	1	
4	Hub	1	
5	以太网线(直线)	1	
6	工件	4	
7	校准用工件	8	
8	校正用指示针	2	

续表

序　　号	设　　备	数　　量	说　　明
9	计算机	1	
10	RT ToolBox2	1	机器人软件
11	In-sight Explorer	1	视觉传感器软件

2. 连接

如图 26-1 所示，机器人与计算机、机器人与视觉传感器都通过以太网进行连接。

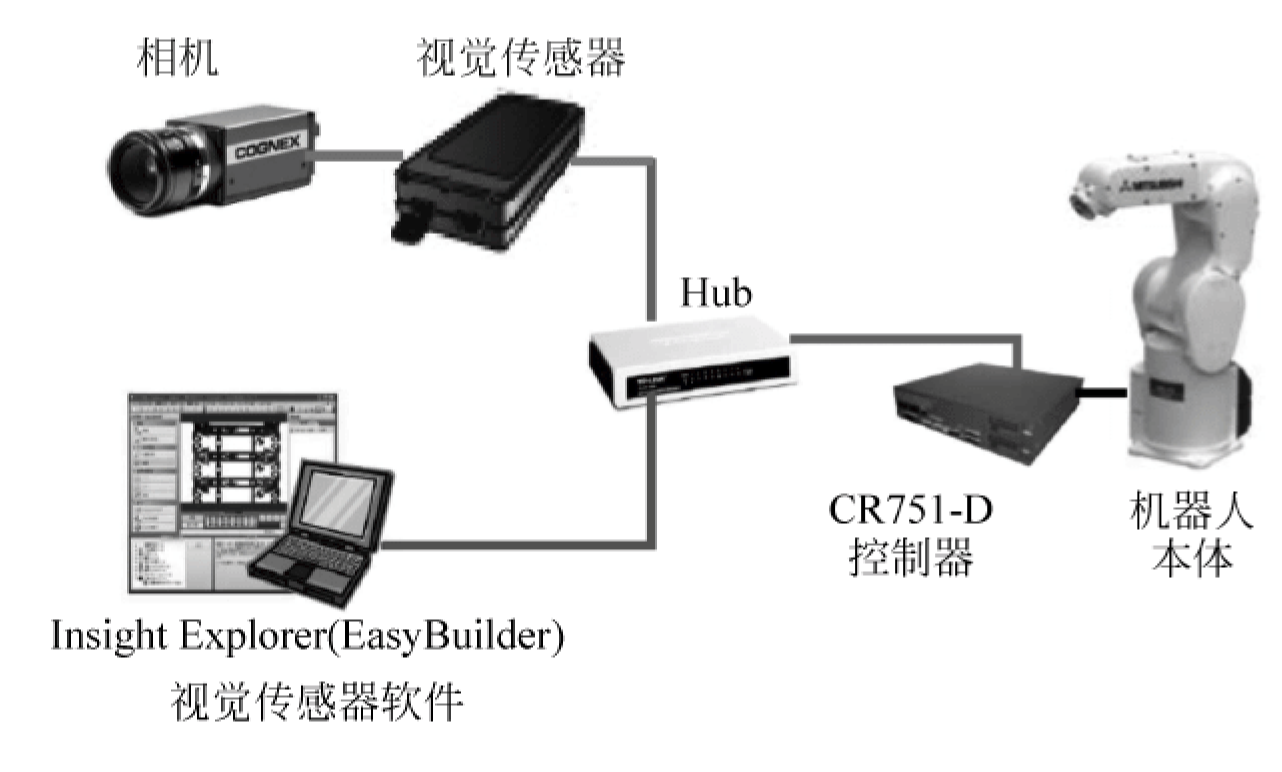


图 26-1　视觉系统与机器人连接

26.2.2　通信设置

由于是以太网通信，设置原则是各通信设备的 IP 地址在同一网段内，即前三位数字相同，第 4 位数字不同。

设置要求如下。

(1) 机器人 IP 地址设置为 192.168.0.20。

(2) 计算机 IP 地址设置为 192.168.0.30，子网掩码设置为 255.255.255。

(3) 视觉传感器 IP 地址设置为 192.168.0.10。

1. 机器人 IP 地址设置

机器人 IP 地址设置如图 26-2 所示。

(1) 在 RT ToolBox2 建立新工作区；

(2) 通信设置选择 TCP/IP；

(3) 单击“详细设定”按钮设置 IP 地址为 192.168.0.20。

单击 OK 按钮，设置完成。

2. 计算机 IP 地址设置

计算机 IP 地址设置如图 26-3 所示。

(1) 单击计算机“控制面板”；

(2) 单击“网络和 Internet 协议”；

(3) 单击“属性”；

(4) 选择 TCP/IPv4；

(5) 设置“IP 地址”为 192.168.0.30，“子网掩码”设置为 255.255.255.0。



图 26-2　机器人 IP 地址设置

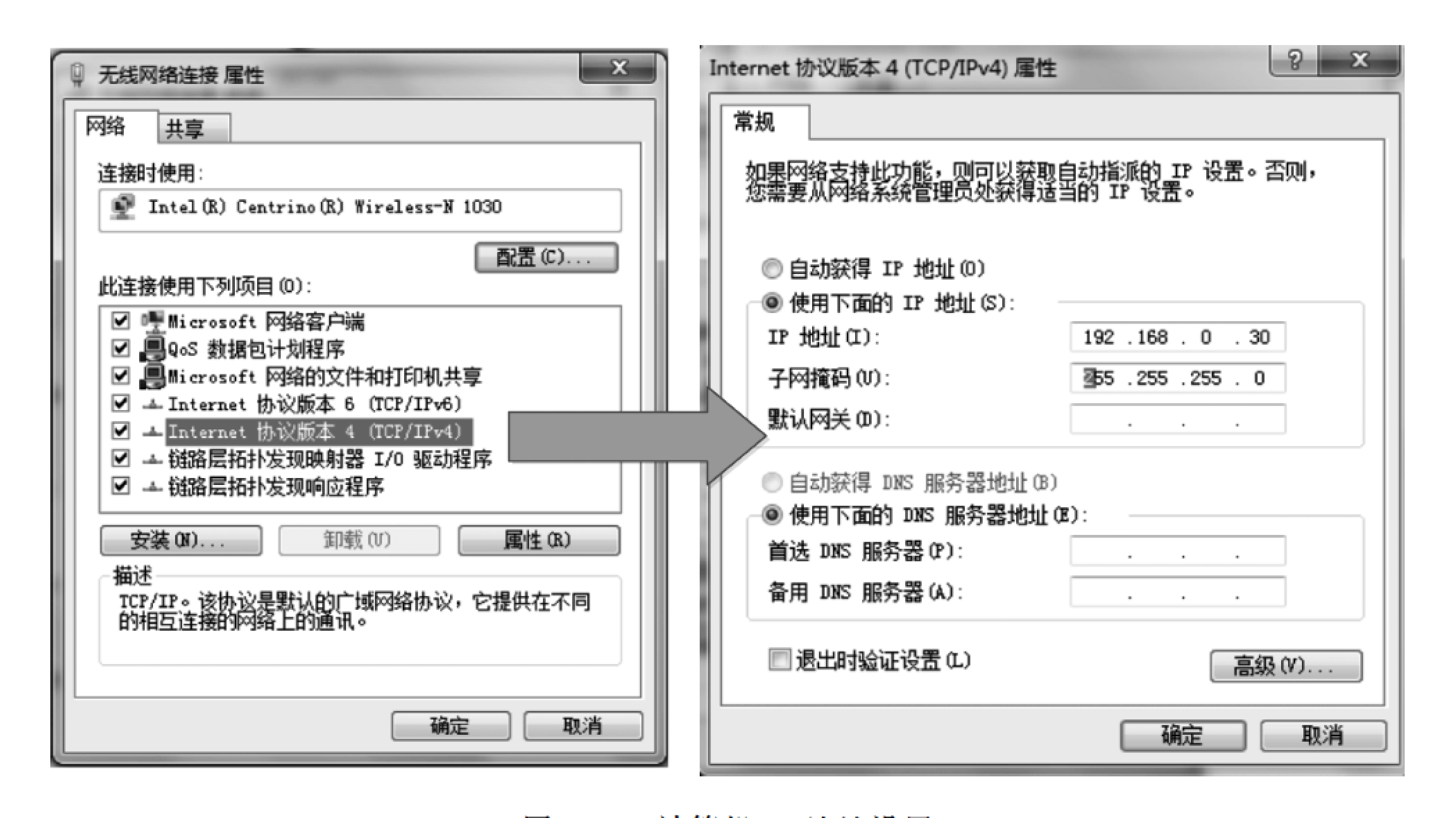


图 26-3　计算机 IP 地址设置

(6) 单击“确定”按钮，设置完成。

3. 视觉传感器 IP 地址设置

操作方法如图 26-4 所示。

(1) 打开 RT 软件，选择“在线”→“参数”→“参数一览”，设定参数 NVTRGTMG=1。

(2) 单击“以太网”→选择“设备及端口”，弹出“设备一览”框。

(3) 在“设备一览”框中选择“OPT19”，弹出“设备参数设定”框。

(4) 在“自动设定”框中选择“网络视觉传感器”。

(5) 在“IP 地址”框中设置“192.168.0.10”。

(6) 在“分配(COMDEV)”框中设置“COM2”作为通信口。

(7) 单击 OK 按钮。

(8) 单击“写入”,设置完成。

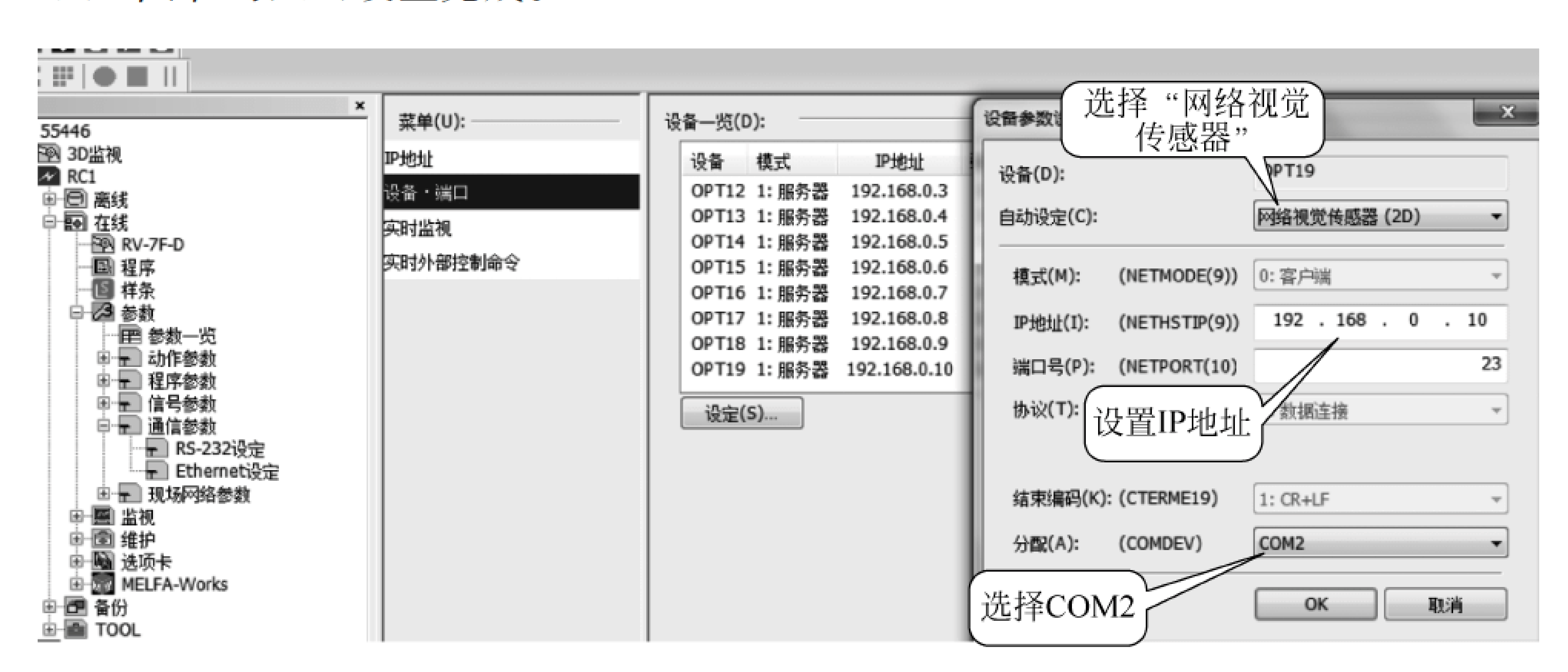


图 26-4　视觉传感器 IP 地址设置

26.3　工具坐标系原点的设置

26.3.1　操作方法

由于在后续的操作中,推荐机器人使用 TOOL 坐标系进行定位,所以必须求出新的 TOOL 坐标系原点。以下是求 TOOL 坐标系原点的方法。

1. 安置指示针

在工作台上安置一个“工件指针”,作为以下机器人抓手校准的标志,如图 26-5 所示。

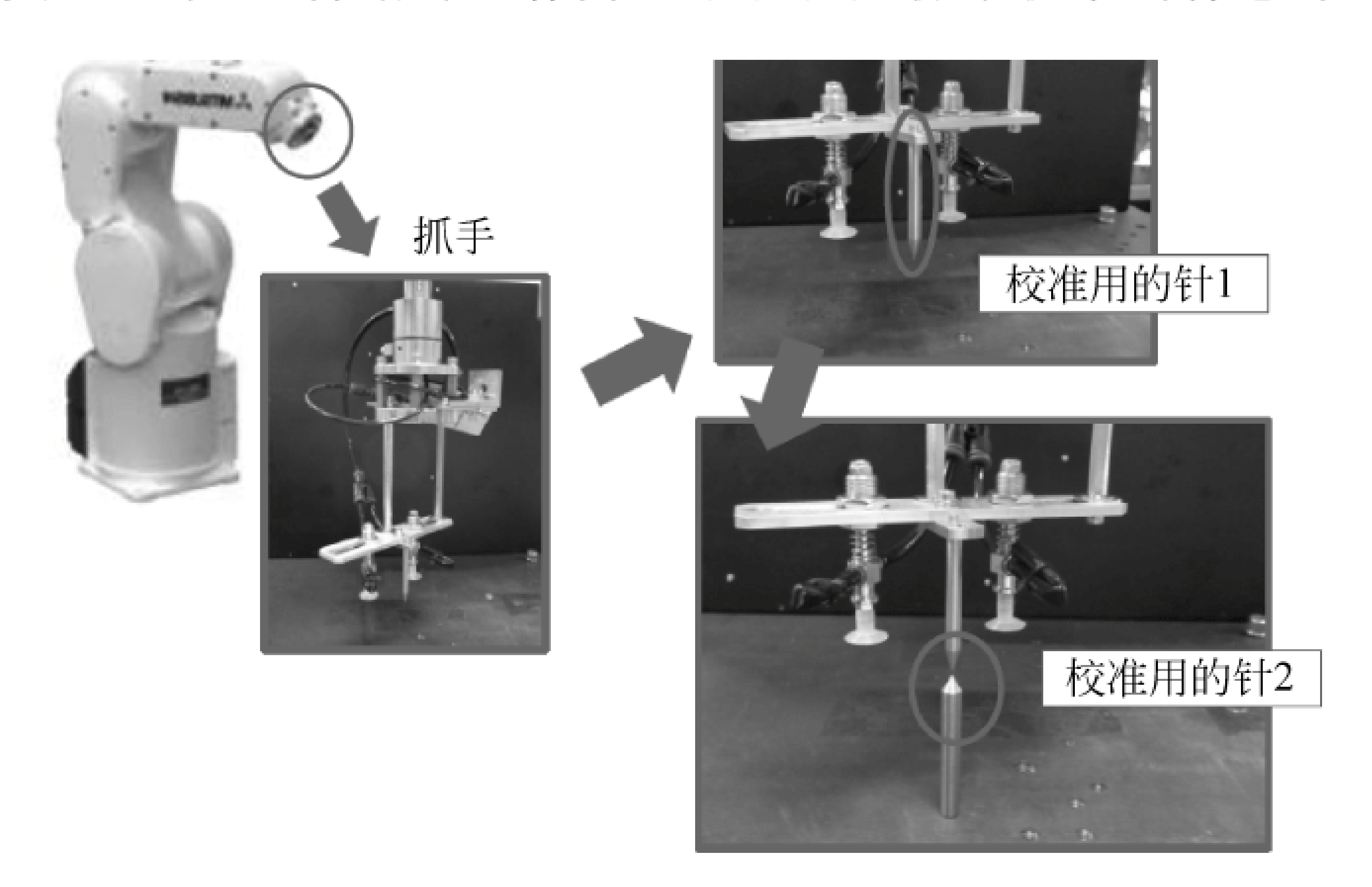


图 26-5　设置 TOOL 坐标系原点

2. 校准方法

(1) 对齐——使用示教单元 JOG 动作,使机器人抓手“指示针尖端”对齐工作台上的“指示针”。

(2) 用示教单元打开程序 TLXY(见 26.3.2 节),按“单步前进”到第 5 行 MVS P91'结束。这时,抓手旋转 90°。

(3) 再次使用示教单元 JOG 动作,使机器人抓手“指示针尖端”对齐桌上指示针。

(4) 用示教单元打开程序 TLXY(见 26.3.2 节),按“单步前进”从第 7 行执行到 END。

(5) 用示教单元 JOG 动作,在“TOOL JOG”模式下,按 C+,C-,机器人动作,当抓手指示针与桌面指示针的相对位置不变时,新的“工具坐标系原点”设置完成。

(6) 在 RT ToolBox2 软件中打开程序 TLXY 和 SVS。

(7) 复制程序 TLXY 中的位置变量 PLT2,粘贴到程序 SVS 中,保存并关闭程序。

26.3.2 求 TOOL 坐标系原点的程序 TLXY

程序 TLXY:

```
'X - YTOOL setting program
PTOOL = P_TOOL' ——P_Tool 为当前设置的 TOOL 坐标系数据。
P0 = P_FBC'—— P_FBC 为编码器反映的当前位置。
P91 = P0 * (0,0,0,0,0,90)'——计算 P91 位置点。
MVS P91'移动到 P91'——绕 Z 轴旋转 90°
1  '以上为第一阶段
2  P90 = P_FBC'——P_FBC 为编码器反映的当前位置。
3  PTL = P_ZERO'——清零。
4  PT = INV(P90) * P0'—— PT 为 P90 与 P0 两点之间的偏差值。重要!
5  PTL.X = (PT.X + PT.Y)/2' ——PTL 为偏差值的中间量。
PTL.Y = ( - PT.X + PT.Y)/2
PLT2 = PTOOL * PTL' ——PLT2 为在原 TOOL 坐标系原点加上偏差量后的位置点。
TOOL PLT2' ——以 PLT2 为当前 TOOL 坐标系原点。
HLT'——暂停
```

26.4 坐标系标定

坐标系标定就是建立视觉传感器坐标系与机器人坐标系之间的关系。当视觉传感器将像素坐标传到机器人时,机器人能够判定像素坐标在机器人坐标系中的位置。简单地说就是建立两个坐标系之间的关系,就相当于建立工件坐标系与基本坐标系之间的关系。

26.4.1 前期准备

(1) 准备带标记的工件,带标记的工件必须与实际工件高度相同。

(2) 标记工件必须准备 5 个。

26.4.2 坐标系标定步骤

1. Insight Explorer 软件的初步设置

(1) 打开软件 Insight Explorer。

(2) 单击菜单“系统”→“将传感器/设备添加到网络”。

(3) 单击弹出的“视觉传感器”,设定“视觉传感器”的 IP 地址为 192.168.0.10,“子网掩

码”为 255.255.255.0。

(4) 单击菜单“系统”→“选项”。

(5) 单击“用户界面”,选择“对 Easy Builder 使用英文符号标记”。

(6) 单击“确定”按钮。

2. 视觉传感器的观察与调节

(1) 选中视觉传感器,单击“连接”。

(2) 单击“联机”,使作业 JOB 进入联机状态。

(3) 单击工具栏中的“触发”按钮,调整镜头的亮度和焦点,一直调整并触发,直到工件清晰地出现在页面内。

(4) 选中应用程序步骤中的“检查部件”。

(5) 打开“几何工具”,单击“用户定义的点”,单击“添加”,一共需要添加 5 个点。

(6) 移动各工件(用户定义点),使其均匀分布在视觉范围内。

(7) 在选择板中可以看到这 5 个点的“像素坐标”(这是最重要的工作目的)。

3. 视觉传感器的精确调节

(1) 移动工件,单击工具栏上的“触发”按钮,不断移动、触发,使十字线交叉点与工件标记点重合(获得像素坐标)。

(2) 使用示教单元移动机器人 JOG,使抓手指示针与工件标记十字线重合(停止,获得机器人坐标)。

4. 坐标系标定

(1) 打开 RT 软件,单击“维护”→2D VISION Calibration,选择 Calibration1(可以选择 Calibration1～Calibration8 中的任何一个),如图 26-6 所示。

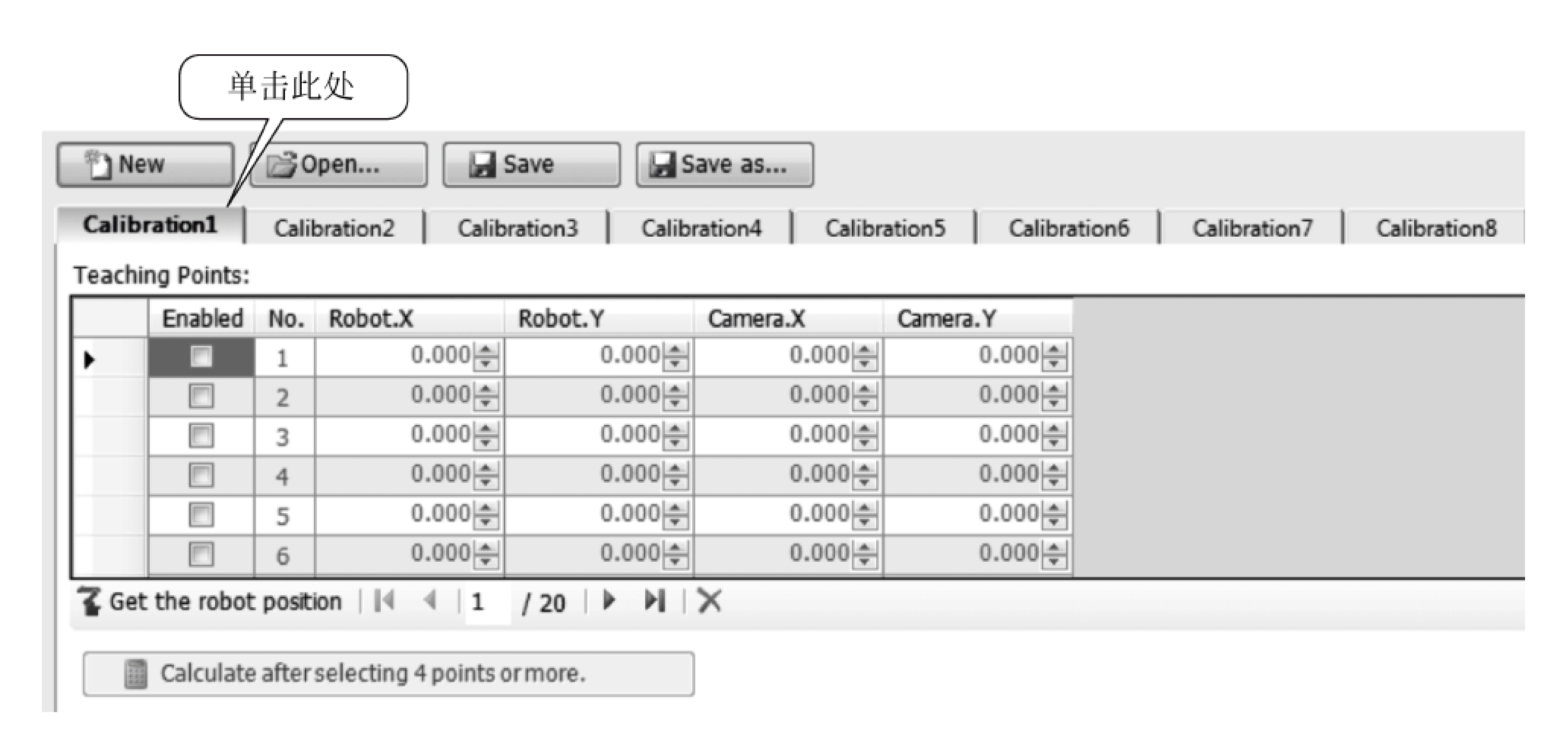


图 26-6　RT 软件设置选择标定序号

(2) 选中做好标记的点(以第 3 点为例),单击 Get the robot position 获得当前位置 X、Y 坐标值,如图 26-7 所示。

(3) 手动输入第 3 点像素坐标。

（4）按照以上方法获得5个点的机器人坐标和像素坐标，在标定完成前，必须一直保持伺服＝ON。

（5）单击 Calculation after selection 4 points or more，计算两个坐标系之间的关系。

（6）单击 Write to robot 按钮写入机器人。

（7）单击 Save 按钮保存。

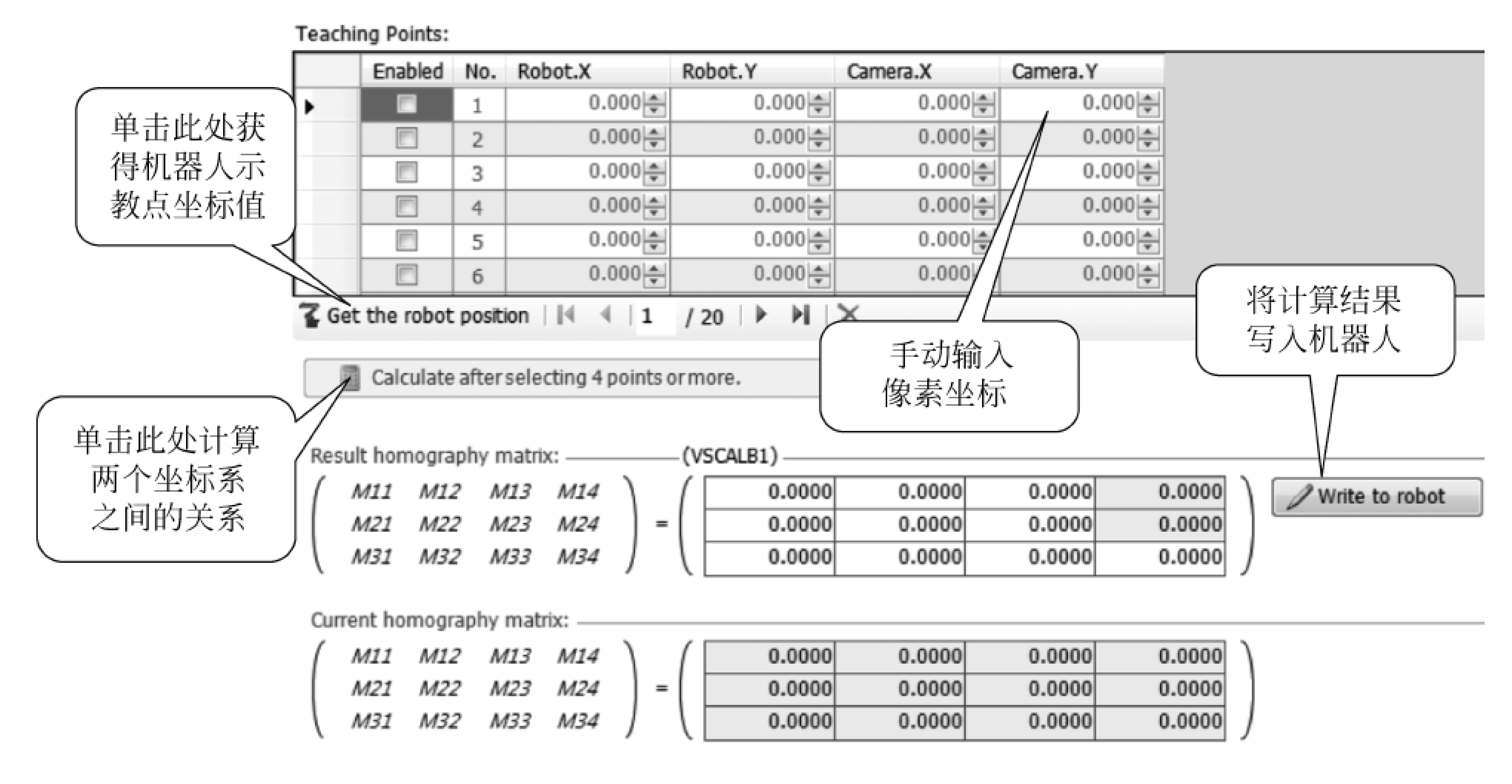


图 26-7　标定方法

经过以上步骤后，标定结束。

26.5　视觉传感器程序制作

定位部件 Pat Max 用于记忆工件的特征，包括工件外部的轮廓和工件上的表面特征，工件周边的颜色即背景色。所以在调整 MODEL 框时，应该尽可能地接近工件的外形，注意周边颜色不能与工件颜色相同，最好颜色反差较大。

制作视觉传感器程序的操作方法如下。

在 Insight Explorer 软件一侧的设置如下。

（1）打开软件 Insight Explorer。

（2）新建作业。

（3）单击“触发”，更新画面。

（4）单击“设置图像”→“触发器”，选择“手动”，调整曝光使图像清晰。

（5）单击“定位部件”，选择“位置工具”中的 Pat Max 图案，单击“添加”按钮。视区内会出现两个矩形，分别是“模型”和“搜索”。

（6）调整“模型”矩形，使其包括工件。

（7）调整“搜索”矩形，使其在视觉传感器的搜索区域内。单击 OK 按钮。

（8）保存本程序，程序名为“TSC.”。

26.6　视觉传感器与机器人的通信

在 Insight Explorer 软件一侧的设置如下。

(1) 打开软件 Insight Explorer。

(2) 单击“通信”→“添加设备”。

(3) 在“设备选择”框中选择“机器人”。

(4) 在“制造商”栏中选择 MITSUBISHI。

(5) 在协议栏中选择“以太网本机字符串”，单击“以太网本机字符串”，单击“添加”按钮。

(6) 单击“Pattern_1”。

(7) 选择 Fixture. X、Fixture. Y、Fixture. Angle、Pass_Count，单击“确定”按钮。

(8) 通过“上”“下”按钮调整顺序。调整结果与后面的 EBRead 指令中的变量顺序有关。

(9) 设定完成后，单击“保存”按钮。

26.7　调试程序

1. 打开软件 Insight Explorer

单击“联机”按钮，使视觉传感器 JOB 在线(注意如果视觉传感器不在线，执行程序会发生报警，报警代码 8650)。

2. 使用示教单元确定工作点

(1) 使用示教单元打开程序 SVS，使机器人以 JOG 模式动作。示教机器人待机位置 Phom 和工件抓取点 Pwk(此点位于工件上，不同于标记点)。示教完成后不可以移动工件。

(2) 使用示教单元关闭程序 SVS，保存程序。

3. RT ToolBox2 软件的使用

(1) 打开程序 SVS，取消 19 行的注释符号，让 19 行程序生效。

(2) 将程序中 17 行、23 行、57 行的内容改为“TSC. JOB”。关闭并保存程序。

(3) 在自动模式下，使用示教单元选择程序 SVS，启动执行 SVS，程序运行结束后，就获得抓取点与识别点之间的补偿量 PH。

4. 调试程序

调试程序 SVS：

```
1  loadset 1.1'——选择抓手及工件。
2  OADL ON'——对应抓手及工件条件，选择最佳加减速模式的指令。
3  SERVO ON '——伺服 ON。
4  WAIT M_SVO = 1 '——M_Svo = 1，伺服电源 = ON。
5  OVRD M_NOVRD'——速度倍率的初始值(100 %)。
6  SPD M_NSPD'——初始速度(最佳速度控制)。
7  ACCEL 100,100'——加减速度倍率指令(%)，设置加减速度的百分数。
8  BASE P_NBASE'——以基本坐标系的初始位置为"当前世界坐标系"。
```

```
9  TOOL PLT2'——以 PLT2 为 TOOL 坐标系原点。
10 '——open port
'以下判断 1# 文件是否开启,如果没有开启则指令 com2 开启,并一直等待 1# 文件开启完成。
11 IF m_nvopen(1)<>1 then
12 nvopen "com2:"as #1
13 wait m_nvopen(1) = 1
14 endif
15 NVLoad #1,"tsc. job"'——加载 tsc.job 作为 1# 文件。
19  GOSUB * MAKE_PH'——调用子程序 * MAKE_PH。
20  *MAIN'——程序分支标志
21 MOV PHOME'——前进到 PHOME 点。
22 DLY 0.5'——暂停 0.5 秒。
23  NVRUN #1,"tsc.job"'——运行 tsc.job 文件并将 tsc.job 作为#1 文件。
24  EBREAD #1,"""",M1,PVS1'——读 1# 文件。
25  IF M1 = 0 THEN ERROR 9102'——如果 M1 = 0 则报警 9102。
26  DLY 0.5'——暂停 0.5 秒。
27  NVRUN #1"tsc.job"'——运行 tsc.job 文件并将 tsc.job 作为#1 文件。
29  EBREAD #1,"""",M1,PVS0'——读 1# 文件。
```

5. 程序的保存

(1) 自动运行完成后,首先关闭在计算机上的程序。此时,不要保存程序,因为此时操作错误的话,刚刚做的 PH 数据就会丢失。

(2) 在 RT ToolBox2 的“在线”中打开程序 SVS,将 19 行“GOSUB ＊MAKE_PH”变为“注释行”——“' GOSUB ＊MAKE_PH”,关闭并保存程序。

26.8 动作确认

(1) 在自动模式下,使用示教单元执行 SVS 程序,机器人会先移动到 PWK 位置,然后再回到 Phome 位置。

(2) 移动工件位置,再次运行 SVS 程序,确认机器人会先移动到 PWK 位置,然后再回到 Phome 位置。

26.9 与视觉功能相关的指令

常用的与视觉应用相关的指令/状态变量如表 26-2 所示。

表 26-2 视觉功能相关的指令一览表

序号	指令	名称	功能
1	NVLoad	加载视觉程序	将指定的视觉程序加载到视觉传感器
2	NVOpen	连接视觉传感器	连接指定的视觉传感器并登录注册该传感器
3	NVClose	关断视觉传感器通信线路	关断视觉传感器通信线路
4	NVIn	接收视觉传感器的识别信息	接收视觉传感器的识别信息
5	NVPst	启动视觉程序并获取数据	启动指定的视觉程序并获取数据
6	NVRun	启动运行视觉程序	启动运行指定的视觉程序

续表

序号	指令	名称	功　　能
7	NVTrg	指令视觉传感器拍摄图像	指令视觉传感器拍摄图像
8	PVSCL	视觉标定指令	
以下是状态变量			
9	P_NvS1～P_NvS8	以位置数据格式保存视觉传感器的识别信息	位置数据
10	M_NvOpen	通信连接状态变量	表示视觉通信是否连接完成
11	M_NvNum	检测到工件总数的状态变量	表示检测到工件总数
12	M_NvS1～M_NvS8	以数值表示识别信息的变量	数值

NV：Network vision sensor，网络视觉传感器。

26.10　视觉功能指令详细说明

26.10.1　NVOpen 连接视觉传感器

1. 功能

连接视觉传感器并登记注册该视觉传感器。

2. 格式

```
NVOpen "<通信口编号>" AS #<视觉传感器编号>
```

3. 术语

1) <通信口编号>(不能省略)

以与 OPEN 指令同样的方法设置通信口编号 COM **，但不能使用 COM1。COM1 口是 TB 单元的 RS-232 通信专用口。设置范围 COM2～COM8。

2) <视觉传感器编号>(不能省略)

设置与机器人通信口连接的视觉传感器编号。设置范围为 1～8。

4. 样例程序

```
If M_NvOpen(1)<>1 Then    '——判断 1# 视觉传感器是否连接。
NVOpen "COM2:" AS #1 '——将视觉传感器连接到 COM2 口并设置为 1# 传感器。
EndIf
Wait M_NvOpen(1) = 1 '——等待 1# 视觉传感器连接完成。
```

5. 说明

(1) 本指令功能为连接视觉传感器到指定的通信口 COM * 并登记注册该视觉传感器。

(2) 最多可连接 7 个视觉传感器。视觉传感器的编号要按顺序设置，用逗号分隔。

(3) 与 OPEN 指令共同使用时，OPEN 指令使用的通信口号 COM ** 和“文件号”与本指令使用的通信口号 COM ** 和“视觉传感器编号”要合理分配，不能重复。例如：

```
Open "COM1:" AS #1
NVOpen "COM2:" AS #2
NVOpen "COM3:" AS #3
```

如下为错误的样例：

```
1  Open "COM2:" AS #1
2  NVOpen "COM2:" AS #2 '——COM2 口已经被占用。
3  NVOpen "COM3:" AS #1'——"视觉传感器编号"已经被占用。
```

在一台机器人控制器和一台视觉传感器的场合，开启的通信线路不能够大于 1。

(4) 注册视觉传感器需要“用户名”和“密码”，因此需要在机器人参数 NVUSER 和 NVPSWD 中设置“用户名”和“密码”。用户名和密码可以为 15 个字符，是数字 0～9 及 A～Z 的集合。T/B(示教器)仅支持大写字母，所以使用 T/B 时设置用户名和密码必须使用大写字母。购置的网络视觉传感器的用户名是“admin”。密码是“”。因此参数 NVUSER 和 NVPSWD 的预设值为：

```
[NVUSER] = "admin" and [NVPSWD] = ""
```

当使用 MELFA-Vision 更改用户名和密码时，必须更改参数 NVUSER 和 NVPSWD。更改参数后断电上电后参数生效。

注意：如果连接多个视觉传感器到一个机器人控制器，则所有视觉传感器必须使用同样的用户名和密码。

(5) 本指令的执行状态可以用 M_NvOpen 状态变量检查。

(6) 如果在执行本指令时，程序被删除，则立即停止。再启动时，按顺序联机传感器。必须复位机器人程序再启动。

(7) 在多任务工作时使用本指令，有如下限制：不同的任务区的程序中，“通信口编号 COM **”“视觉传感器编号”不能相同。

① 如果使用了相同的 COM ** 编号，则会出现“attempt was made to open an already open communication file(试图打开已经被开启了的一个通信口)”报警。

如图 26-8 所示，在任务区 2 和任务区 3 中都同时指定了 COM2 口，所以报警。

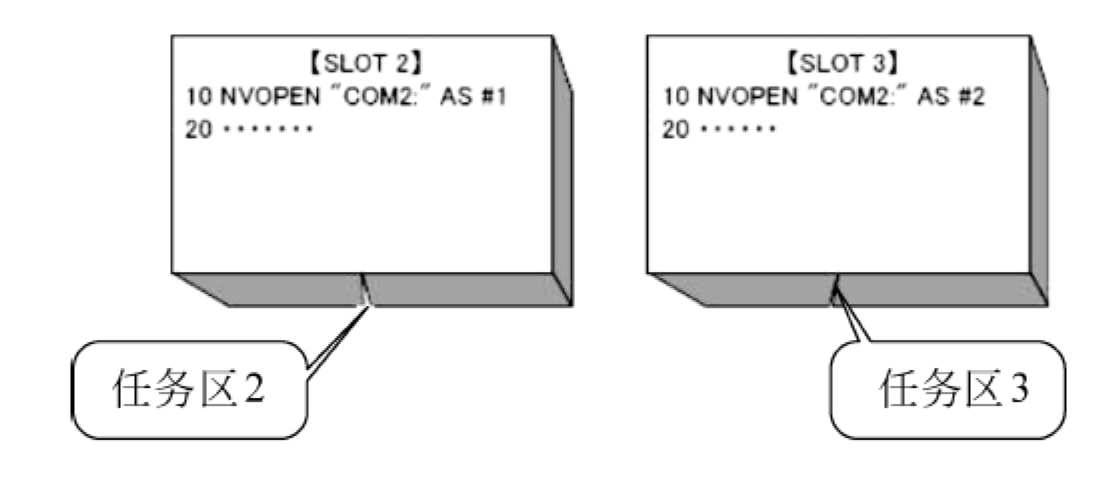


图 26-8 在任务区 2 和任务区 3 中都同时指定了 COM2 口

② 如果设置了同样的“视觉传感器编号”也会报警。

图 26-9 中，在任务区 2 和任务区 3 中都指定了 1# 视觉传感器，所以报警。

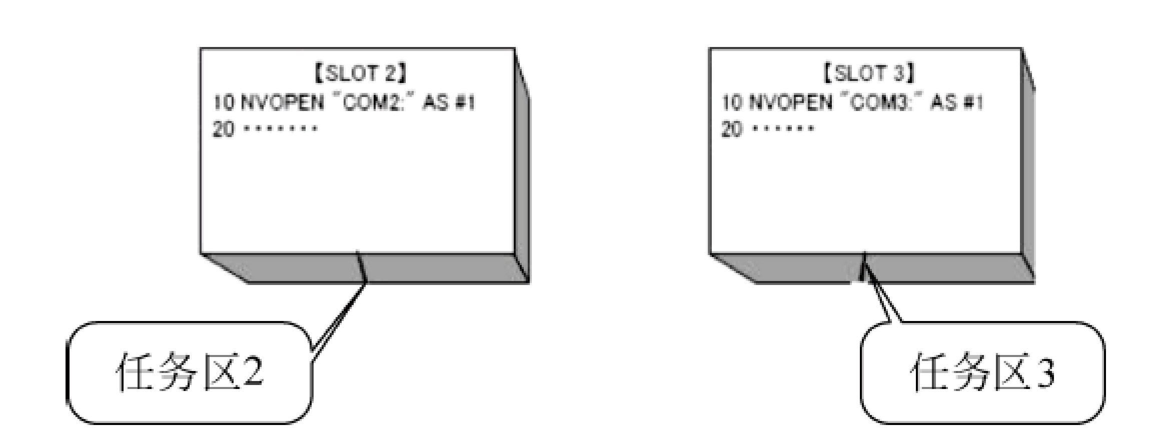


图 26-9 在任务区 2 和任务区 3 中都指定了 1# 视觉传感器

(8) 不支持启动条件为 Always 与设置为“连续功能”的程序。

(9) 在构建系统时要注意,一个视觉传感器可以同时连接三个机器人,如果连接第 4 个机器人,则第 1 个被切断。

(10) 调用子程序时,通信连接不会被切断,但是主程序的 END 指令与复位指令会切断通信连接。

如果在执行本指令时,某个中断程序的启动条件成立,则立即执行中断程序。

6. 报警

(1) 如果数据类型是错误的,则会出现“syntax error in input command(指令语法错误)”报警。

(2) 如果 COM ** 口编号不是 COM2～COM8,会报警。

(3) 如果视觉传感器编号不是 1～8,会报警。

26.10.2 NVClose——关断视觉传感器通信线路指令

1. 功能

关断视觉传感器通信线路。

2. 格式

```
NVClose[[ # ]<视觉传感器编号>[, [[ # ]<视觉传感器编号> … ]
```

3. 术语

视觉传感器编号(不能省略):指连接到机器人通信口的视觉传感器编号(可能在一个网络上有多个视觉传感器)。设置范围为 1～8。如果有多个传感器,用逗号分隔。

4. 样例程序

```
1  If M_NvOpen(1)<> 1 Then '——判断 1# 视觉传感器是否联机完成。
2  NVOpen "COM2:" AS #1 '——在 COM2 口连接视觉传感器并将其设置为 1# 传感器。
3  EndIf
4  Wait M_NvOpen(1) = 1'——等待 1# 传感器联机通信完成。
5  …
10 NVClose #1'——关断 1# 视觉传感器与 COM2 口的通信。
```

5. 说明

(1) 本指令功能为关断在 NVOpen 指令下的通信连接。

(2) 如果省略了视觉传感器编号,则切断所有视觉传感器的通信连接。

(3) 如果通信线路已经切断,则转入下一步。

(4) 由于可以同时连接 7 个视觉传感器,所以必须按顺序编写视觉传感器编号,这样可以按顺序关断视觉传感器。

(5) 如果执行本指令时程序被删除,则继续执行本指令直到本指令处理的工作完成。

(6) 如果在多任务中使用本指令,在使用本指令的任务中,仅需要关闭由 NVOpen 指令打开的通信线路。

(7) 不支持启动条件为 Always(上电启动)和设置为“连续功能”的程序。

(8) 如果使用 END 指令,所有由 NVOpen 或 Open 指令开启的连接都会被切断。但是在调用子程序指令下不会关断。程序复位也会切断通信连接,所以在程序复位和 END 指

令下不使用本指令也会切断通信连接。

(9) 如果在执行本指令时,有某个中断程序的启动条件已经成立,则在执行完本指令后才执行中断程序。

6. 报警

如果“视觉传感器编号”超出 1～8 范围,则会出现“超范围报警。

26.10.3 NVLoad——加载程序指令

1. 功能

加载指定的视觉程序到视觉传感器。

2. 格式

```
NVLoad #<视觉传感器名称>,"<视觉程序名称>"
```

3. 术语

<视觉程序名称>——指要启动的视觉程序名称(已经存在的视觉程序名称可以省略),只可以使用 0～9,A～Z,a～z,“-”,以及下画线“_”,对程序进行命名。

4. 样例程序

```
If M_NvOPen(1)<> 1 Then '——判断 1#视觉传感器是否联机完成。
NVOpen "COM2:" AS #1 '——在 COM2 口连接视觉传感器并将其设置为 1#传感器。
EndIf
Wait M_NvOpen(1) = 1 '——等待通信连接完成。
NVLoad #1,"TEST" '——加载 Test 程序。
NVPst #1, "","E76","J81","L84",0,10'——启动#1 程序并获取信息。
```

5. 说明

(1) 本指令功能为加载指定的程序到指定的视觉传感器。

(2) 在加载程序到视觉传感器的位置点,本指令将移动到下一步。

(3) 如果执行本指令时删除了程序,立即停机。

(4) 如果指定的程序名已经被加载,则本指令立即结束不做其他处理。

(5) 在执行多任务时使用本指令,必须在任务区执行 NVOpen 指令,同时必须用 NVOpen 指令指定传感器编号。

(6) 不支持启动条件为 Always(上电启动)与设置为“连续功能”的程序。

(7) 如果在执行本指令时,某个中断程序的启动条件成立,则立即执行中断程序。

26.10.4 NVPst——启动视觉程序获取信息指令

1. 功能

启动指定的视觉程序并获取信息。从视觉传感器接收的数据存储于机器人控制器作为状态变量。

2. 格式

```
NVPst #<视觉传感器编号>,"<视觉程序名称>", "<存储识别工件数据量的单元格号>", "<开始单元
格编号>", "<结束单元格编号>", <数据类型> [, <延迟时间>]
```

3. 术语

(1) <视觉传感器编号>：对使用的视觉传感器设置的编号(不能省略)，设置范围 1～8。

(2) <视觉程序名称>(不能省略)：设置视觉程序名称。已经加载的视觉程序可省略。只有 0～9，A～Z，a～z，“-”，以及下画线“_”等字符可以使用。

(3) <存储识别工件数据量的单元格号>：指定一个单元格。在这个单元格内存储被识别的工件数量。设置范围：行 0～399，列 A～Z，例如 A5。

被识别的“工件数”存储在 M_NvNum(*)(* =1～8)。

(4) <开始单元格编号>/<结束单元格编号>(不能省略)：指定(电子表格内)视觉传感器识别信息的存放范围(从起始到结束)。单元格的内容存储在 P_NvS * (30)、M_NvS * (30，10)、C_NvS * (30，10)(* =1～8)等变量中。

设置范围：行 0～399，列 A～Z，例如 A5，如图 26-10 所示。

但是，当指定的行为 30，列为 10，或单元格总数超过 90 时就会出现“设置的单元格数超出范围”报警。

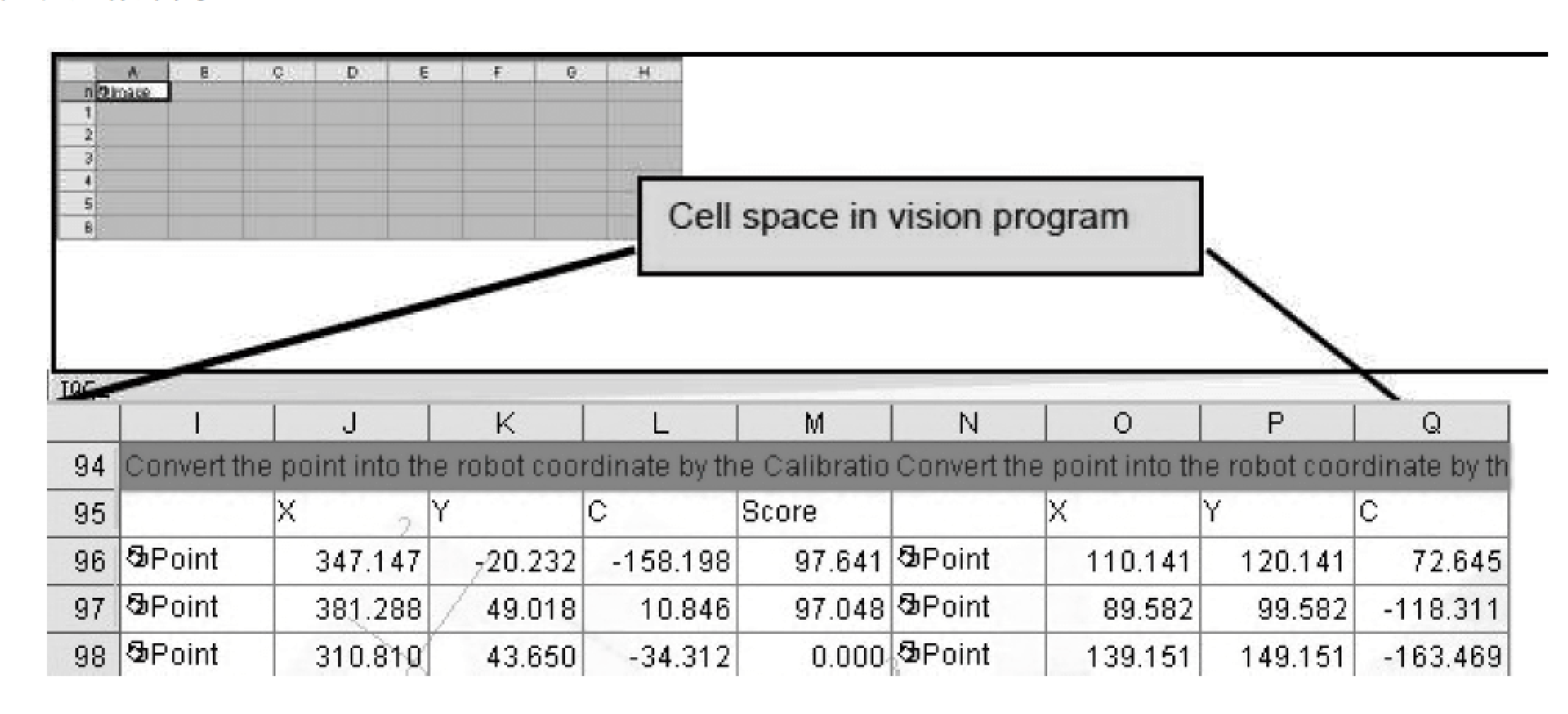


图 26-10　在视觉程序内的电子表格及单元格

(5) <数据类型>(不能省略)：用于设置所获取的数据类型。所获取的数据类型有位置型数据、单精度实数、文本型数据。设置范围为 0～7。具体设置见表 26-3。

表 26-3　数据类型设置表

设定值	0	1	2	3	4	5	6	7
单元格状态	一个数据/单元格(每个单元格内放一个数据)				(每个单元格内放两个或更多个数据)			
对应使用的状态变量	P_NVS()	M_NVS()	C_NVS()	M_NVS() C_NVS()	P_NVS()	M_NVS()	C_NVS()	M_NVS() C_NVS()
数据类型	位置型数据	单精度实数	文本型	单精度实数 文本型	位置型数据	单精度实数	文本型	单精度实数 文本型

(6) <延迟时间>：本指令执行的时间。

4. 样例程序

```
If M_NvOpen(1)<> 1 Then
NVOpen "COM2:" AS #1
```

```
3  EndIf
4  Wait M_NvOpen(1) = 1 .
5  NVPst #1,"TEST","E76","J81", "L84",1,10
'——启动运行 TEST 程序，在 E76 单元格内存放识别工件数量。识别信息存放区域为 J81～L84,"数据类型"为"单精度实数"。同时识别信息还存放在机器人状态变量 M_NvS1()中。
30 NVclose #1 '——关闭通信线路
```

5. 说明

(1) 本指令功能为启动视觉程序并接收识别信息。

(2) 在延迟时间内，直到信息接收完成之前，不要移动到下一步。

注意：在机器人程序停止时，本指令立即被删除。程序重新启动后，继续处理。

(3) 如果指定的程序已经被加载，本指令无须加载程序而立即执行，可以缩短处理时间。

(4) 当在多任务状态下使用本指令时，必须使用 NVOpen 指令。

(5) 如果 Type 设置为 4～7，则可以提高信息接收的速度。

(6) 不支持启动条件为 Always(上电启动)和设置为“连续功能”的程序。

(7) 如果在本指令执行过程中，有任一中断程序执行条件成立，则立即执行中断程序。

6. 多通道模式的使用方法

当使用多通道模式时，根据机器人的数量设置启动单元格和结束单元格以取得信息，同时“数据类型”设置为 1～3。

下例是一个多通道模式的信息处理方法。

如图 26-11 所示，设置启动单元格和结束单元格为 J96～M98，则给第一个机器人的信息存储在视觉程序表格 J97～M98 中。

	I	J	K	L	M	N	O	P	Q
94	Convert the point into the robot coordinate by the Calibratio					Convert the point into the robot coordinate by th			
95		X	Y	C	Score		X	Y	C
96	Point	347.147	-20.232	-158.198	97.641	Point	110.141	120.141	72.645
97	Point	381.288	49.018	10.846	97.048	Point	89.582	99.582	-118.311
98	Point	310.810	43.650	-34.312	0.000	Point	139.151	149.151	-163.469

图 26-11　视觉程序电子表格信息

传送给第二个机器人的信息存储在视觉程序表格 O97～R98 中。如果在 NVPst 指令中设置<数据类型>=1，则数据被存储在 M_NVS1()中，如表 26-4 所示。

表 26-4　M_NVS1()中的实际数据

Row		Column								
		1	2	3	4	5	6	7	8	9
M_NVS1()	1	347.147	−20.232	−158.198	97.641	0.0	0.0	0.0	0.0	0.0
	2	381.288	49.018	10.846	97.048	0.0	0.0	0.0	0.0	0.0
	3	310.81	43.65	−34.312	0.0	0.0	0.0	0.0	0.0	0.0
	4	0.0	0.0	0.0	0.0	0.0	0	0	0	0
	5	0.0	0.0	0.0	0.0	0.0	0	0	0	0

例如，如果为 2 通道模式，<启动单元格>＝J96，<结束单元格>＝R98，<数据类型>＝1，则存储在 M_NVS1(30,10)，结果如表 26-5 所示。

表 26-5　2 通道模式中 M_NVS1 的数据

Row		Column								
		1	2	3	4	5	6	7	8	9
M_NVS1()	1	347.147	－20.232	－158.198	97.641	0.0	110.141	120.141	72.645	97.641
	2	381.288	49.018	10.846	97.048	0.0	89.582	99.582	－118.311	97.048
	3	310.81	43.65	－34.312	0.0	0.0	139.151	149.151	－163.469	95.793
	4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

一台视觉传感器最多可同时连接三台机器人控制器，不过本指令在同一时间只能使用一次。本指令可以用于任一机器人控制器。

7. 三台机器人与一台视觉传感器构成的跟踪系统实例

工作步骤如下，如图 26-12 所示。

(1) 三台机器人，一台设置为主站，主站使用 NVPST 指令向视觉传感器发出“拍照请求”，视觉传感器启动拍照，当拍照结束后，将数据信息传送到机器人主站。

(2) 主站机器人发出“接收通知”给另外两台机器人(推荐两台机器人之间使用 I/O 连接，另一台机器人使用以太网连接)。

(3) 使用 NVIN 指令，每台机器人可分别接收各自的信息。

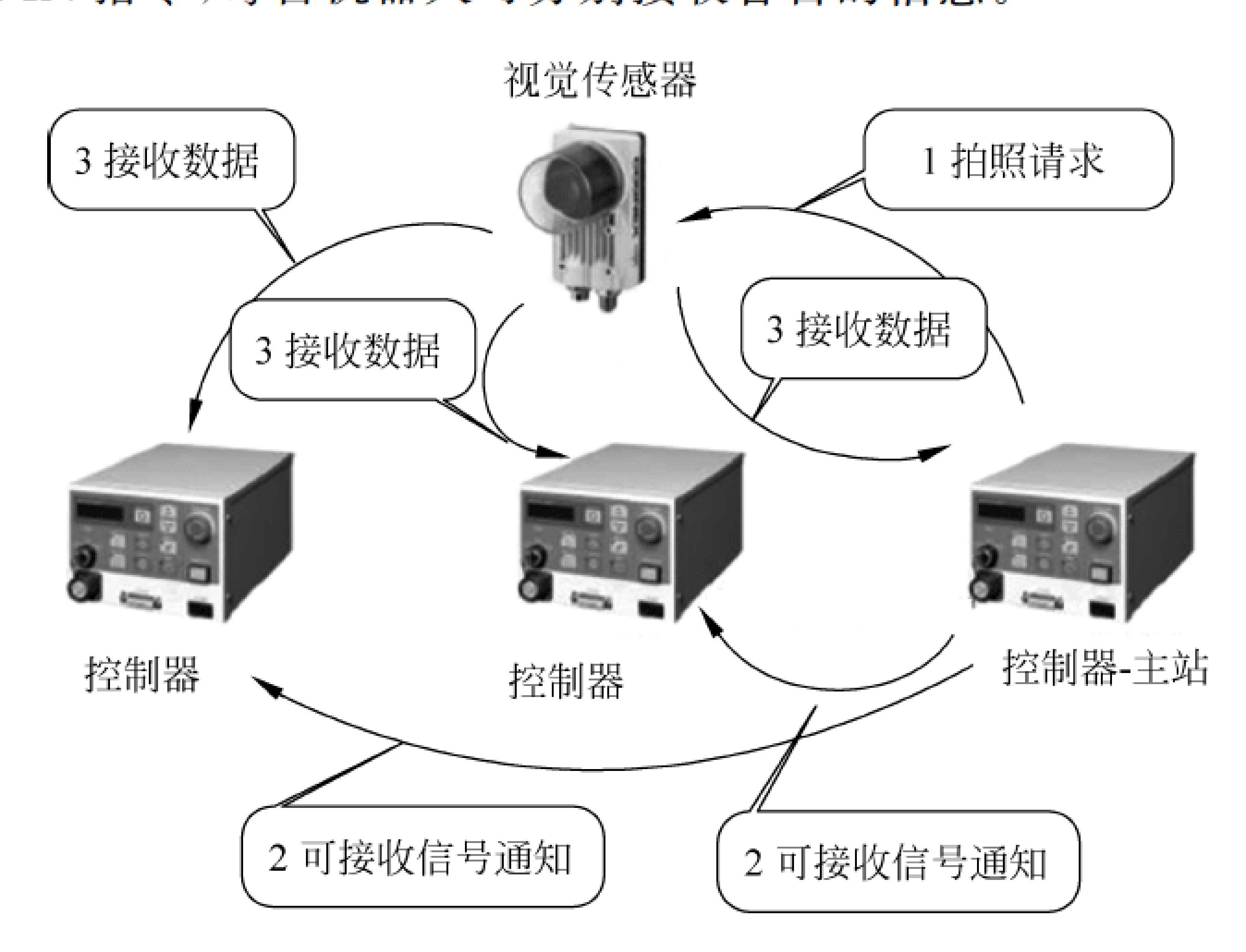


图 26-12　三台机器人与一台视觉传感器构成的系统

8. 两台机器人与一台视觉传感器构建系统的样例

工作步骤如下，如图 26-13 所示。

(1) 当前使用视觉传感器的控制器首先要检查视觉传感器没有被另一台控制器使用并向另外一台控制器发出“在使用中”的信号。

(2) 向视觉传感器发出“拍照请求”。

(3) 当视觉传感器处理完成图像数据后，控制器就接收必要的数据。

(4) 控制器关闭"在使用中"的信号并输出给另外一控制器。

(5) 另一台控制器执行步骤(1)~(4)。

用这种方法,两台控制器能够交替使用一台视觉传感器。

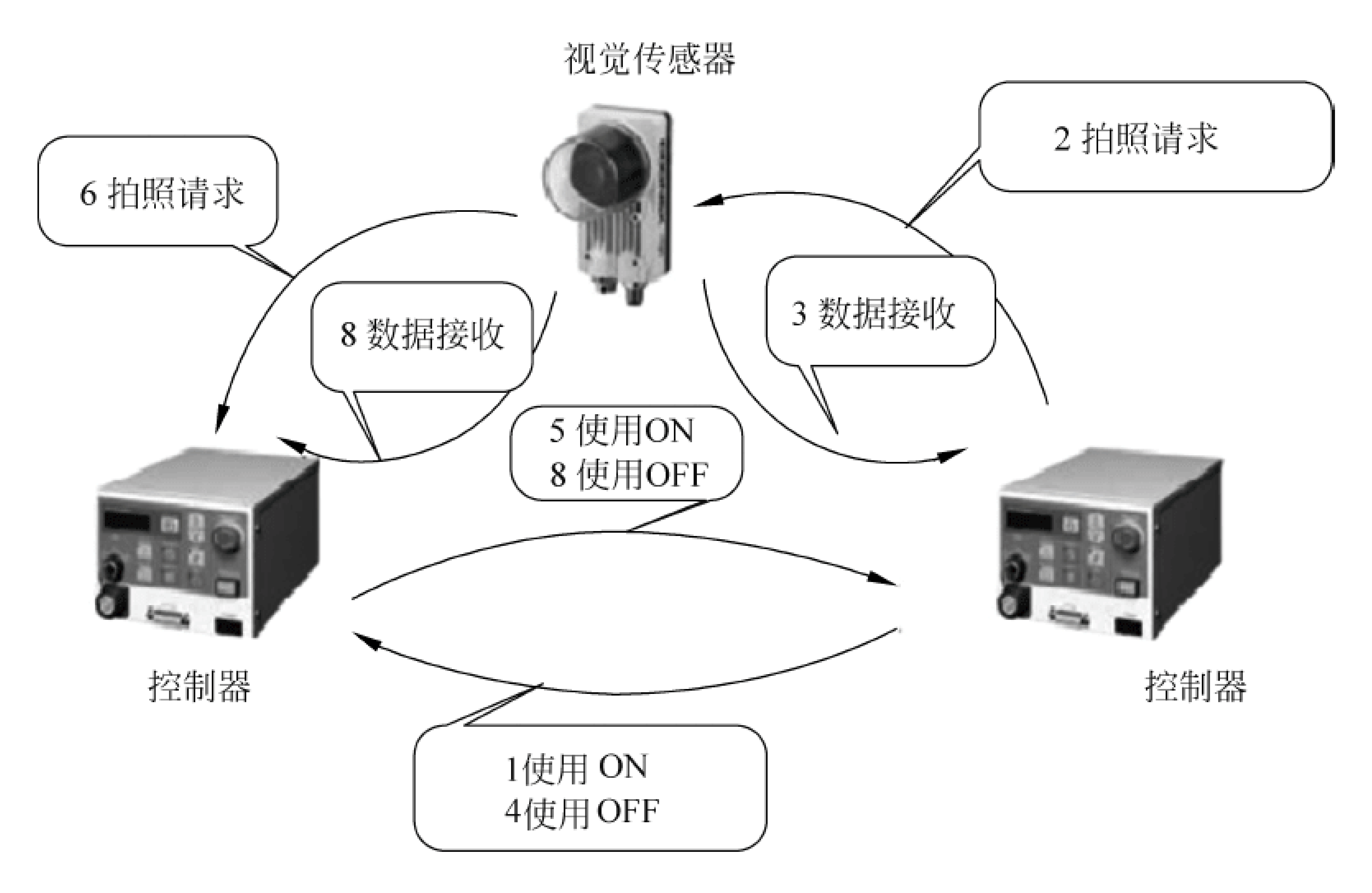


图 26-13　两台机器人与一台视觉传感器构成的系统

26.10.5　NVIn——读取信息指令

1. 功能

接收来自视觉传感器的识别信息,这些识别信息被保存在机器人控制器中作为状态变量。

2. 格式

NVIn＃<视觉传感器编号>,"<视觉程序名称>", "<存储识别工件数据量的单元格号>", "<开始单元格编号>", "<结束单元格编号>", <数据类型> [, <延迟时间>]

3. 样例程序

```
1  If M_NvOpen(1)<> 1 Then '——判断 1＃ 传感器是否联机完成,如果没有联机完成,就连接到"COM2:"。
2  NVOpen "COM2:" AS ＃1
3  EndIf
4  Wait M_NvOpen(1) = 1
5  NVRun ＃1,"TEST" '——启动 TEST 程序。
6  NVIn ＃1, "TEST","E76","J81","L84",0,10'——
接收信息。在 E76 内存放"工件数量"。J81～L84 存放识别的数据。数据是"位置变量"。"位置变量"存储在 P_NvS1(30)中。
30 NVClose ＃1 '——关闭＃1 文件。
```

4. 说明

(1) NVIn 指令与 NVPST 指令的区别在于：NVIn 指令仅仅是一个读取信息的指令。而 NVPST 指令是先启动程序运行再读取信息的指令。NVIn 指令与 NVPST 指令的术语定义完全相同。

(2) 通过设置数据类型,将读取的信息存放在 P_NvS1(30)中。

26.10.6 NVRun——视觉程序启动指令

1. 功能

启动运行指定的程序。

2. 格式

```
NVRun ＃<传感器编号>,"<传感器程序名>"
```

3. 样例程序

```
If M_NvOpen(1)<>1 Then '——判断 1＃ 传感器是否联机完成,如果没有联机完成就连接到"COM2:"。
NVOpen "COM2:" AS ＃1 '——将传感器连接到通信口 COM2。
EndIf
Wait M_NvOpen(1)=1 '——等待联机完成。
NVRun ＃1,"TEST" '——启动运行 TEST 程序。
NVIn 1, "TEST","E76","J81","L84",0,10 '——输入相关数据。
```

26.10.7 NVTrg——请求拍照指令

1. 功能

向视觉传感器发出拍照请求。

2. 格式

```
NVTrg ＃<视觉传感器编号>,<延迟时间>, <存放 1＃编码器数据的状态变量> [,[<存放 2＃编码器数据的状态变量>][,[<<存放 3＃编码器数据的状态变量>] [,[<<存放 4＃编码器数据的状态变量>][,[<存放 5＃编码器数据的状态变量>]
        [,[<存放 6＃编码器数据的状态变量>][,[<存放 7＃编码器数据的状态变量>] [,[<存放 8＃编码器数据的状态变量>]
```

3. 术语

(1) < Delay time >延迟时间：从向传感器发出拍照指令到编码器数据被读出的时间。设置范围 0～150ms。

(2) <存放 N＃编码器数据的状态变量>：指定一个双精度变量。这个变量存储从外部编码器 n 读出的数据，n＝1～8。

4. 样例程序

```
If M_NvOpen(1)<>1 Then '——判断 1＃ 传感器是否联机完成,如果没有联机完成就连接到"COM2:"。
NVOpen "COM2:" AS ＃1
EndIf
Wait M_NvOpen(1)=1 '——等待联机完成。
NVRun ＃1,"TEST" '——运行 TEST 程序。
NVTrg ＃1,15,M1＃,M2＃ ——'请求 1＃视觉传感器拍照并在 15ms 后将编码器 1 和编码器 2 的值存储在变量 M1＃和 M2＃中。
```

26.10.8 P_NvS1～P_NvS8——位置型变量

1. 功能

P_NvS＊是以位置数据格式保存的视觉传感器的识别信息。在 NVPst 指令或 NVIn

指令中，设置< type >=0 时，数据以 X，Y，C 坐标数据的格式保存识别信息，也就是"位置型数据"。

样例如图 26-14 所示。

	I	J	K	L	M	N	O	P	Q
94	Convert the point into the robot coordinate by the Calibratio					Convert the point into the robot coordinate by th			
95		X	Y	C	Score		X	Y	C
96	Point	347.147	-20.232	-158.198	97.641	Point	110.141	120.141	72.645
97	Point	381.288	49.018	10.846	97.048	Point	89.582	99.582	-118.311
98	Point	310.810	43.650	-34.312	0.000	Point	139.151	149.151	-163.469

图 26-14　视觉程序信息表格

在 NVPst 指令或 NVIn 指令中，设置<启动单元格>=J96，<结束单元格>=L98，则 P_NVS1()如下（有三点有效的数据）。

```
P_NvS1(1) = ( + 347.14,  - 20.23,  + 0.00,  + 0.00,  + 0.00,  - 158.19,  + 0.00,  + 0.00)(0, 0)
P_NvS1(2) = ( + 381.28,  + 49.01,  + 0.00,  + 0.00,  + 0.00,  + 10.84,  + 0.00,  + 0.00)(0, 0)
P_NvS1(3) = ( + 310.81,  + 43.65,  + 0.00,  + 0.00,  + 0.00,  - 34.312,  + 0.00,  + 0.00)(0, 0)
P_NvS1(4) = (  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00)(0, 0)
P_NvS1(5) = (  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00)(0, 0)
     ⋮
P_NvS1(30) = (  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00,  + 0.00)(0, 0)
```

要注意：在 P_NvS1 * 位置数据中，没有 Z、A、B 及第 7 轴，第 8 轴数据，直接使用 P_NvS1 * 要特别注意检查。注意程序样例中的处理方法。

2. 格式

```
<位置 变量> = P_NVS * (<位置点编号>)
```

* (1～8)：视觉传感器编号。

3. 术语

<位置点编号>：1～30。

4. 样例程序

```
If M_NvOpen(1)<> 1 Then '——如果 1# 传感器通信连接未完成。
NvOpen "COM2:" AS #1'——将 1# 传感器连接到 COM2 通信口。
EndIf
Wait M_NvOpen(1) = 1 '——等待 1# 传感器通信连接完成。
NvPst #1,"TEST","E76","J96","L98",0,10 '——读取的数据存放在 P_NvS1 ()。
MvCnt = M_NvNum(1) '——以"检测到的工件数量"设置为 MvCnt。
For MCnt = 1 To MvCnt '——以"检测到的工件数量"编制一个循环程序。
P10 = P1        '——赋值。
P10 = P10 * P_NvS1(MCnt) '——将 P10 * P_NvS1(MCnt)相乘获得完整位置信号。
Mov P10, - 50 '——前进到 P10 点的近点。
Mvs P10 '——前进到 P10 点。
HClose 1 '——抓手动作。
Mvs P10, - 50 '——前进到 P10 点的近点。
Next MCnt'——循环语句。
```

26.10.9　M_NvNum ——状态变量

1. 功能

存储视觉传感器检测到的工件数量(0～255)。

2. 格式

```
<数字变量> = M_NVNUM (<传感器编号>)
```

3. 术语

<数字变量>：设置需要使用的数字变量。

4. 样例程序

```
If M_NvOpen(1)<>1 Then '——如果1#传感器通信连接未完成。
NvOpen "COM2:" AS #1 '——将1#传感器连接到COM2通信口。
EndIf
Wait M_NvOpen(1)=1 '——等待联机完成
NvPst #1,"TEST","E76","J81","L84",1,10'——运行TEST程序并读信息。
MvCnt=M_NvNum(1) '——将1#视觉传感器检测到的工件数量赋值到MvCnt。
```

5. 说明

(1) M_NvNum是一个状态变量,用于表示执行NVPst指令或NVIn指令时,视觉传感器检测到的工件数量。

(2) 存储的识别数量数据会一直保存,直到执行下一个NVPst指令或NVIn指令。

26.10.10　M_NvOpen——状态变量

1. 功能

存储视觉传感器通信连接状态。在NVOpen指令之后,检查通信连接是否连接完成。

M_NVOPEN=0——未连接完成;

M_NVOPEN=1——连接完成;

M_NVOPEN=-1——未连接。

2. 格式

```
<数字变量> = M_NVOPEN (<视觉传感器编号>)
```

3. 样例程序

```
If M_NvOpen(1)<>1 Then '——判断:如果1#视觉传感器没有连接。
NvOpen "COM2:" AS #1 '——将视觉传感器连接到COM2口并设置为1#视觉传感器。
EndIf
Wait M_NvOpen(1)=1 '——等待连接完成。
```

4. 说明

初始值M_NvOpen=－1,在执行NVOpen指令时,M_NvOpen=0,表示通信线路未连接完成,随后M_NvOpen=1,表示联机完成。

26.11.11 M_NvS1～M_NvS8——视觉传感器识别的数值型变量

1. 功能

M_NvS＊是状态变量。M_NvS＊以数字格式存储视觉传感器的识别数据。在 NVPst 指令或 NVIn 指令中，如果设置< type >=1,3,5,7，则其指定范围内的识别信息被转换为数值并存储。

在图 26-15 上部，是视觉程序所获得的识别信息。使用 NVPst 指令或 NVIn 指令，设置<启动单元格>=J96,<停止单元格>=L98。

这样在状态变量 M_NvS＊中，其对应的数据如图 26-15 下部所示。

	I	J	K	L	M	N	O	P	Q
94	Convert the point into the robot coordinate by the Calibratio					Convert the point into the robot coordinate by th			
95		X	Y	C	Score		X	Y	C
96	Point	347.147	-20.232	-158.198	97.641	Point	110.141	120.141	72.645
97	Point	381.288	49.018	10.846	97.048	Point	89.582	99.582	-118.311
98	Point	310.810	43.650	-34.312	0.000	Point	139.151	149.151	-163.469

	Element 2 / Element 1	1	2	3	4	5	6	7	8	9
M_NVS1()	1	347.147	-20.232	-158.198	97.641	0.0	110.141	120.141	72.645	0.0
	2	381.288	49.018	10.846	97.048	0.0	89.582	99.582	-118.311	0.0
	3	310.81	43.65	-34.312	0.0	0.0	139.151	149.151	-163.469	0.0
	4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

图 26-15 视觉识别信息与变量的关系

2. 格式

```
<数字变量> = M_NVS＊ (<行编号>, <列编号>)
```

＊（1～8）：视觉传感器编号。

注意：M_NvS＊与 P_ NvS＊的区别在于，P_ NvS＊是一个“位置型变量”，可以表示一个位置点。而 M_NvS＊只是表示电子表格中一个单元格的数据。请注意样例程序中的用法。

3. 样例程序

```
If M_NvOpen(1)<>1 Then
NvOpen "COM2:" AS ＃1
EndIf
Wait M_NvOpen(1)=1
NvPst ＃1,"TEST","E76","J96","Q98"0.1,10 '——数据类型=1,获取的数据存放在 M_NVS1()。
MvCnt=M_NvNum(1) '——获取视觉传感器检测到的工件数。
For MCnt=1 To MvCnt '——循环。
P10=P1          '——赋值计算。
P10.X=M_NvS1(MCnt,1) '——指定 M_NvS1()第 1 行第 1 列的数据 = P10.X。
P10.Y=M_NvS1(MCnt,2) '——指定 M_NvS1()第 1 行第 2 列的数据 = P10.Y。
P10.C=M_NvS1(MCnt,3) '——指定 M_NvS1()第 1 行第 3 列的数据 = P10.C。
Mov P10,-50 '——前进到 P10 点的近点。
```

```
13 Mvs P10 '——前进到 P10 点。
14 HClose 1 '——抓手动作。
15 Mvs P10,-50 '——前进到 P10 点的近点。
16 Next MCnt'——下一循环。在下面的循环中,MCnt=2,3,4。
```

4. 说明

(1) M_NvS* 中的数据会一直保持直到执行下一个 NVPst 指令或 NVIn 指令。但是在断电、执行 END 指令、程序 reset 指令时,M_NvS* 的数据会被清零。

同时,如果<数据类型>设置不是 1,3,5,7,M_NvS* 的数据会被清零。

(2) 如果识别的数据是字符串,M_NvS* 的数据会被清零。

26.10.12 EBRead——读数据指令(康奈斯专用)

1. 功能

读出指定视觉传感器的数据。这些数据被存储在指定的变量中。本指令专用于康奈斯公司的 EB 软件制作的视觉程序。

2. 格式

```
EBRead #<视觉传感器编号>,[<标签名称>],<变量 1>[,<变量 1 2>]…[,<延迟时间>]
```

3. 术语

视觉程序名:指定视觉程序名。读出该程序内存储的数据。如果省略视觉程序名,则要在参数 EBRDTAG 内设置。初始值为"Job.Robot.FormatString"。

延迟时间:设置范围 1~32 767s。

4. 样例程序

```
100 If M_NvOpen(1)<>1 Then
110   NVOpen "COM2:" As #1
120 End If
130 Wait M_NvOpen(1)=1'——等待联机完成。
140 NVLoad #1,"TEST" '——加载 TEST 程序。
150 NVRun #1,"TEST" '——运行 TEST 程序。
160 EBRead #1,,MNUM,PVS1,PVS2 '——读出程序名为 Job.Robot.FormatString 内的数据。将这些数据存储在指定的变量中。
```

5. 说明

(1) 本指令用于读取数据。

(2) 数据存储在指定的变量中。

(3) 变量用逗号分隔,数据按变量排列的顺序存储,因此读出数据的类型要与指定变量的类型相同。

(4) 当指定变量为位置型变量时,存储的数据为 X、Y、C,而其他各轴的值为“0”。C 值的单位为弧度。

(5) 当指定的变量数少于接收数据时,则接收的数据仅存储在指定的变量中。

(6) 当指定的变量数多于接收数据时,则多出的数据部分不上传。

(7) 如果省略视觉程序名称则默认为参数 EBRDTAG 的初始值 Job.Robot.FormatString。

(8) 在延迟时间内不要移动到下一步，必须等到数据读出完成。注意，如果机器人程序停止，本指令立即被删除，需要用重启指令继续执行本指令。

(9) 多任务时必须使用 NVOpen 指令和 NVRun 指令，在相关的任务区程序内指定视觉传感器的编号。

(10) 不支持启动条件为“上电启动”和连续功能的程序。

如果执行本指令时，某一中断程序的条件成立，则立即执行中断程序。待中断程序执行完毕后再执行本指令。

为了缩短生产时间，可以在执行 NVRun 指令和 EBRead 后执行其他动作。

如果在执行 NVRun 指令后立即执行 EBRead 指令，必须设置参数 NVTRGTMG ＝1。

如果参数 NVTRGTMG ＝出厂值，则在执行 NVRun 指令后的下一个程序无须等待视觉程序的处理完成即可执行。

如果在 NVRun 和 EBRead 之间，程序停止，则执行 NVRun 指令和 EBRead 指令的结果可能不同。

6. 指令样例

变量值：执行 EBRead 指令的变量如下。

(1) 如果视觉程序的内容是 10，则：

① 执行“EBRead ＃1,"Pattern_1. Number_Found",MNUM 指令后，MNUM＝10。

② 执行“EBRead ＃1,"Pattern_1. Number_Found",CNUM”指令后，CNUM＝10。

(2) 如果执行视觉程序 Job. Robot. FormatString 的内容为：

```
2, 125.75, 130.5, -117.2, 55.1, 0, 16.2,
```

① 执行“EBRead ＃1,,MNUM,PVS1,PVS2”后，MNUM＝2。

```
PVS1.X = 125.75   PVS1.Y = 130.5   PVS1.C =- 117.2
PVS2.X = 55.1     PVS2.Y = 0       PVS2.C = 16.2
```

其他轴数据＝0。

注意：PVS1,PVS2 是位置型变量，所以读出的数据为“位置点数据”。

② 执行“EBRead ＃1,,MNUM,MX1,MY1,MC1,MX2,MY2,MC2”后，MNUM＝2。

```
MX1 = 125.75   MY1 = 130.5   MC1 =- 117.2
MX2 = 55.1     MY2 = 0       MC2 = 16.2
```

注意：MX1,MY1,MC1 是数据型变量，所以读出的数据为“数字”。

③ 执行“EBRead ＃1,,CNUM,CX1,CY1,CC1,CX2,CY2,CC2”后，CNUM＝"2"。

```
CX1 = "125.75"   CY1 = "130.5"   CC1 = " - 117.2"
CX2 = "55.1"     CY2 = "0"       CC2 = "16.2"
```

注意：CX1,CY1,CC1 是字符串型变量，所以读出的数据为“字符串。

(3) 如果执行视觉程序 Job. Robot. FormatString 的内容为

```
2, 125.75, 130.5
```

则执行“EBRead ＃1,,MNUM,PVS1”后，MNUM＝2。

```
PVS1.X = 125.75   PVS1.Y = 130.5
```

其他轴数据＝0。

26.11　需要思考的问题

(1) 机器人和视觉系统通过什么方式连接？如何设置 IP 地址？
(2) 如何使用 RT 软件进行视觉标定？
(3) 如何使用 NVOpen 指令？
(4) 如何使用 NVPST 指令？
(5) NVIN 指令有什么特点？
(6) 如何使用 EBread 指令？

第 27 章　第 27 日——机器人与视觉系统联合应用

【学习目的】

在第 26 章学习了机器人与视觉系统联合应用的基本理论知识和相关指令。内容较多，本章学习实际的应用案例，可结合第 26 章的内容进行学习。

27.1　案　例　1

1. 工况状态及要求

在图 27-1 中，工件放置的“标准位置”的缺口是垂直方向(设置为 0°位置)。同时要求经过机器人搬运到下一工位后，工件的放置位置仍然为缺口为 0°的位置。但是工件的实际上料位置可能千变万化(缺口位置可能是任意角度)，所以用视觉系统获得工件的实际上料位置，其工作流程如图 27-2 所示。

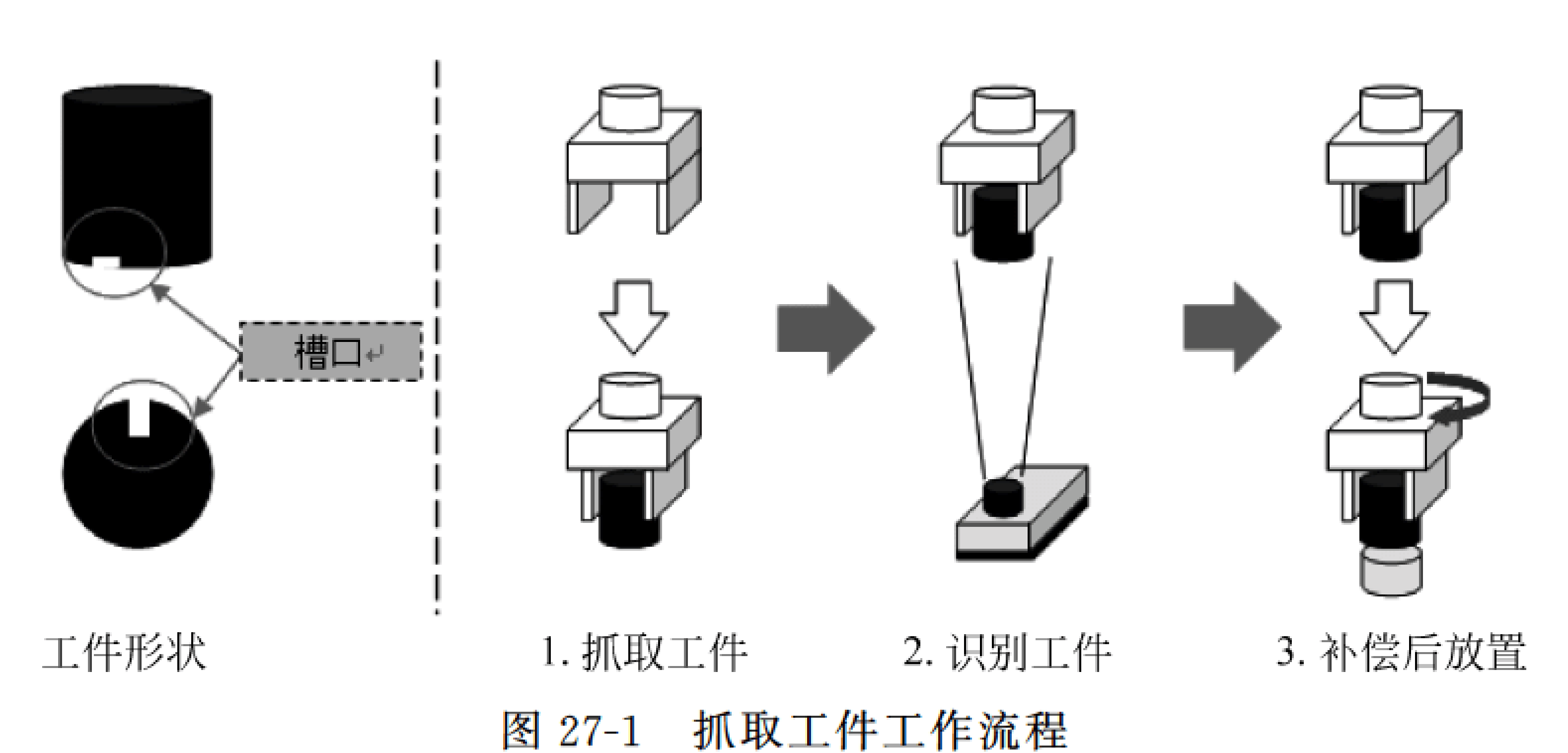


图 27-1　抓取工件工作流程

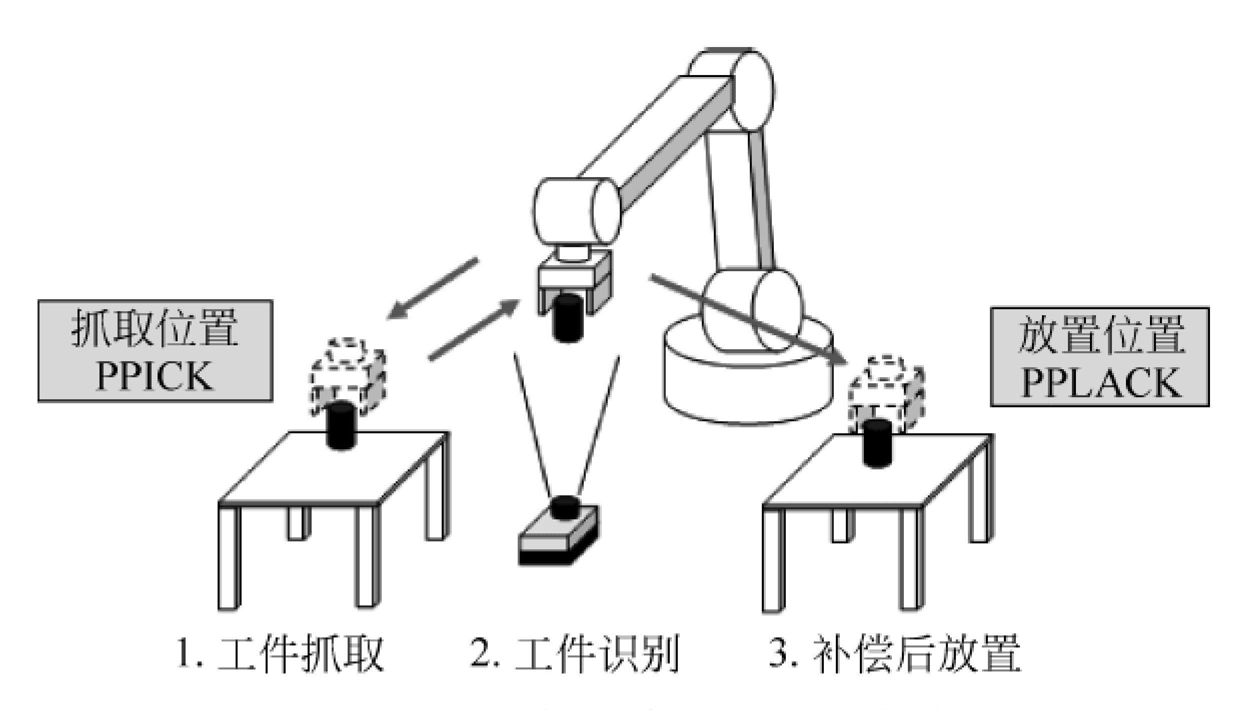


图 27-2　抓取工件工作流程中的各位置点

2. 解决问题的思路

在图 27-1 中,工件放置的标准位置的缺口是垂直放置的,但是实际上料位置可能千变万化。解决问题的思路是使用视觉相机拍摄被抓取工件的实际位置。求出标准位置数据与实际位置数据的偏差量,将该偏差量作为后续定位位置的补偿量。

在这种方法中,关键是如何计算偏差量。图 27-3 是计算偏差量的示意图。其实"标准位置数据 PVSCHK"和"视觉相机拍摄获得的数据 PVSO"都是以机器人坐标系确定的数据。通过空间位置点的计算就可以获得这两个位置点之间的"偏差量"(PH=Inv(PVS0)×PVSCHK)。

视觉传感器识别区域

图 27-3　计算"偏差量"示意图

在图 27-2 和图 27-3 中,各点位的意义如下。

PVSCHK——拍摄位置的标准位置点。用 JOG 示教获得。

PVS0——拍摄位置的实际位置点。用相机拍摄获得。

PH——偏差量。计算获得。

PPICK——抓料点位置。

PPLACE——下一工位放置点的位置。

3. 操作方法

(1) 在相机拍摄位置先用 JOG 方式标定标准位置的坐标。

(2) 运行视觉程序获得工件的实际位置。

(3) 编写程序计算偏差量。

(4) 在最后的放置工件位置补偿这个偏差量。这样就得到了正确的放置工件位置。

4. 拍照获取数据并计算偏差量程序

根据图 27-2 和图 27-3,步骤如下。

步骤 1:

```
1  Mov PHOME'——移动到待机位置。
2  Mov PPICK, -50'——移动到抓料位置上方。
3  Mvs PPICK'——移动到抓料位置。
4  Dly 0.5'——暂停 0.5 秒。
5  HClose 1 '——抓手 = ON。
6  Dly 0.5'——暂停 0.5 秒。
7  Mvs PPICK, -50'——移动到 PPICK 点的近点。
8  Mov PVSCHK'——移动到拍照位置。
9  Dly 0.5'——暂停 0.5 秒。
10 Hlt '——暂停。
```

步骤 2:

```
11 If M_NvOpen(1) <> 1 Then'——判断是否联机通信。
12 NVOpen "COM2:" As #1'——设置"COM2:"为通信口,并将从"COM2:"输入的文件作为#1 文件。
13 Wait M_NvOpen(1) = 1'——等待联机完成。
14 EndIf'——判断语句结束。
```

```
15 NVRun #1,"sample.job"'——运行视觉程序。
16 EBRead #1,,MNUM,PVS '——读取视觉识别信息。
17 If MNUM = 0 Then * VS_MISS'——如果无法识别就跳转到报警程序。
18 PVS0 = P_Zero'——对 PVS0 清零。
19 PVS0 = PVS '——设置 PVS0 为识别点位置。
20 PVS0.FL1 = PVSCHK.FL1'——设置 PVS0 的位置结构标志与标准位置相同。
21 PH = Inv(PVS0) * PVSCHK'——计算偏差量。重要!
22 Hlt '——暂停。
```

步骤 3：

```
P_01 = PVSCHK'——设置各有关点为全局变量。
23 P_02 = PH'—— 注意这是偏差量。
24 P_03 = PPLACE'——放置位置。
25 P_04 = PHOME'——原点位置。
26 P_05 = PPICK'——抓取点位置。
27 Hlt'——暂停。
28 End'——程序结束。
29 '***** ERROR 报警子程序 *****
30 * VS_MISS'——程序分支标志。
31 Error 9001'——报警 9001。
32 GoTo * VS_MISS'——跳转到 * VS_MISS 行。
33 Return'——子程序结束。
```

5. 说明

(1) NVRun 命令紧接着为 EBRead 命令时，必须设置参数 NVTRGTMG＝1。

如果参数 NVTRGTMG 为出厂设定(NVTRGTMG＝2)时，NVRun 命令不等待视觉识别处理结束就执行下一条命令，因此紧接着执行 EBRead 命令的话，有可能会读出上一次的识别结果。

(2) 执行 EBRead 命令时，识别结果(识别工件数)保存至变量 NMUM 中，工件的识别位置(X、Y、C)保存至变量 PVS 中。识别结果 NG 或识别数量为 0 时，会进行报警处理。

(3) 在识别的角度数据的符号颠倒的情况下，请追加如下处理。

```
PVS0.C = PVS.C * ( - 1)
```

(4) 由于步骤 1 中使用的变量需要传送给步骤 2，使用了全局变量。本程序中，使用 P_01～P_05。全局变量在本工程的所有程序中都生效，使用时务必注意。

6. 应用偏差量进行补偿的搬送程序

```
1  PVSCHK = P_01'——代入相关变量。
2  PH = P_02'——代入相关变量。
3  PPLACE = P_03'——代入相关变量。
4   PHOME = P_04'——代入相关变量。

5  PPICK = P_05'——代入相关变量。
6  Mov PHOME'——移动到待机点。
7  Dly 0.5'——暂停 0.5 秒。
8  HOpen 1'——抓手张开。
9  If M_NvOpen(1) <> 1 Then'——判断通信连接状态。
```

```
 NVOpen "COM2:" As ＃1'——设置"COM2:"为通信口,并将从"COM2:"输入的文件作为＃1 文件。
Wait M_NvOpen(1) = 1'——等待连接完成。
EndIf'——判断语句结束。
Mov PPICK, - 50'——移动到取料点上方。
 Mvs PPICK'——移动到取料点。
Dly 0.5'——暂停 0.5 秒。
HClose 1'——抓手闭合。
Dly0.5'——暂停 0.5 秒。
Mvs PPICK, - 50'——移动到取料点上方。
Mov PVSCHK'——移动到拍照位置。
Dly0.5'——暂停 0.5 秒。
NVRun ＃1,"sample.job"'——运行视觉程序。
 EBRead ＃1,,MNUM,PVS'——读出视觉识别信息＊1。
 If MNUM = 0Then ＊VS_MISS'——如果无法识别就跳转到报警程序。
 PVS1 = P_Zero'——对 PVS1 进行初始化设置。
PVS1 = PVS＊2'——对 PVS1 进行设置。
PVS1.FL1 = PVSCHK.FL1'——对 PVS1 进行设置。
PPLACE1 = PPLACE＊PH'——关键!对工件放置位置进行补偿(位置点乘法运算)。
Mov PPLACE1, - 50'——移动到放置点上方。
Mvs PPLACE1'——移动到放置点。
Dly0.5'——暂停 0.5 秒。
HOpen1'——抓手张开。
Dly0.5'——暂停 0.5 秒。
Mvs PPLACE1, - 50'——移动到放置点上方。
Mov PHOME'——移动到 PHOME 点。
Hlt'——暂停。
End
'＊＊＊＊＊ ERROR ＊＊＊＊＊
＊VS_MISS'——程序分支标志。
Error9001'——报警 9001。
GoTo ＊VS_MISS   '——跳转到"＊VS_MISS"行。
Return  '——子程序结束。
```

7. 说明

(1) 视觉程序 Job 的识别结果(识别工件数)保存至变量 NMUM 中,将工件的识别位置(X,Y,C)保存至变量 PVS 中。识别结果 NG 或识别数量为 0 时,会进行报警处理。

(2) 在识别的角度数据的符号颠倒的情况下,需要追加如下处理。

```
PVS0.C = PVS.C＊(- 1)
```

8. 操作方法

步骤 1:

(1) 在工件识别的面贴上画有十字线的纸,然后使用机器人抓取工件。

(2) 设定工具点使纸上画的十字线的交点成为旋转中心。

(3) 机器人抓住工件向拍照位置移动,使工件上的十字线与相机镜头对焦。对焦完成后的位置即为位置变量 PVSCHK。

(4) 进行位置变量 PVSCHK 确认的同时,实施视觉传感器的 N 点校准(2D 校准)。

(5) 打开抓手取出工件,设置工件抓取位置。

（6）机器人抓取出工件的位置即为位置变量 PPICK。

（7）将待机位置设置为位置变量 PHOME。

（8）执行自动运行，运行至 Hlt 停止。

步骤 2：

（1）使用示教单元打开程序，JOG 运行使十字线再次对焦，再次示教位置变量 PVSCHK。

（2）对焦完成后制作工件识别 Job 并保存。

（3）使视觉传感器在线，自动运行至停止。运行视觉程序并执行读取指令 EBREAD 后的数据如图 27-4 所示。

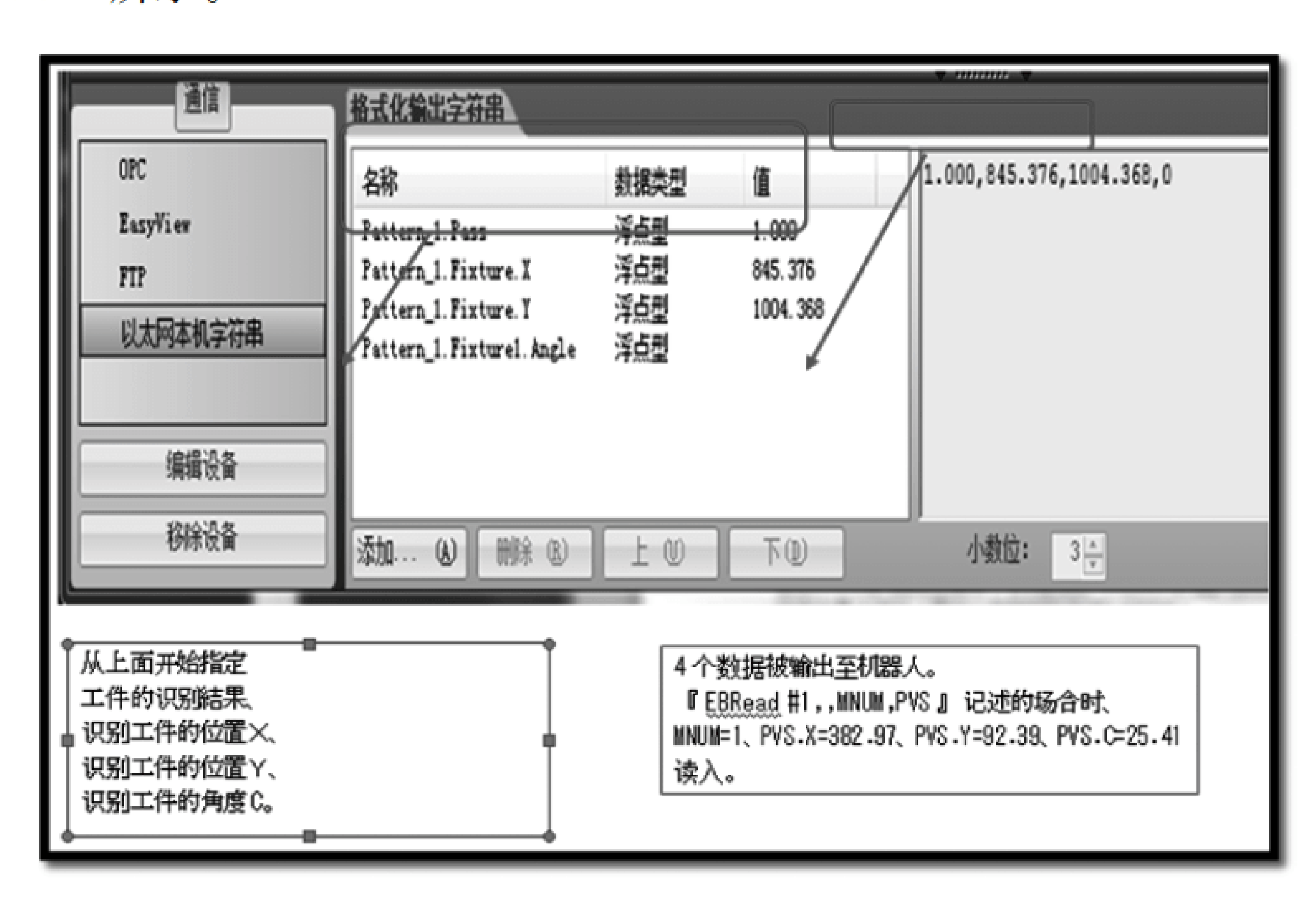


图 27-4　视觉程序信息

步骤 3：

（1）使用 TB 打开程序，通过 JOG 运行将工件的放置位置示教为位置变量 PPLACE。

（2）放开工件，将机器人移动至待机位置，保存程序。

27.2　案　例　2

1. 工况要求

如图 27-5 所示，工作要求为在工件 A 上安装工件 B，由机器人夹持工件 B 安装在工件 A 上。

工件 A 由其他设备上料，每次工件 A 的实际位置与理想基准位置都会发生偏差，所以要靠视觉照相机获得实际的位置信息。

在图 27-5 中有几个重要的位置点，现说明如下。

P_20——基准标志点。工件 A 在理想基准位置的照相标志点。

PA1——基准安装工作点。工件 A 在理想基准位置的安装工作点。

PLN——实际标志点。工件 A 在实际位置的照相标志点。

PACT——实际安装工作点。工件 A 在实际位置的安装工作点。

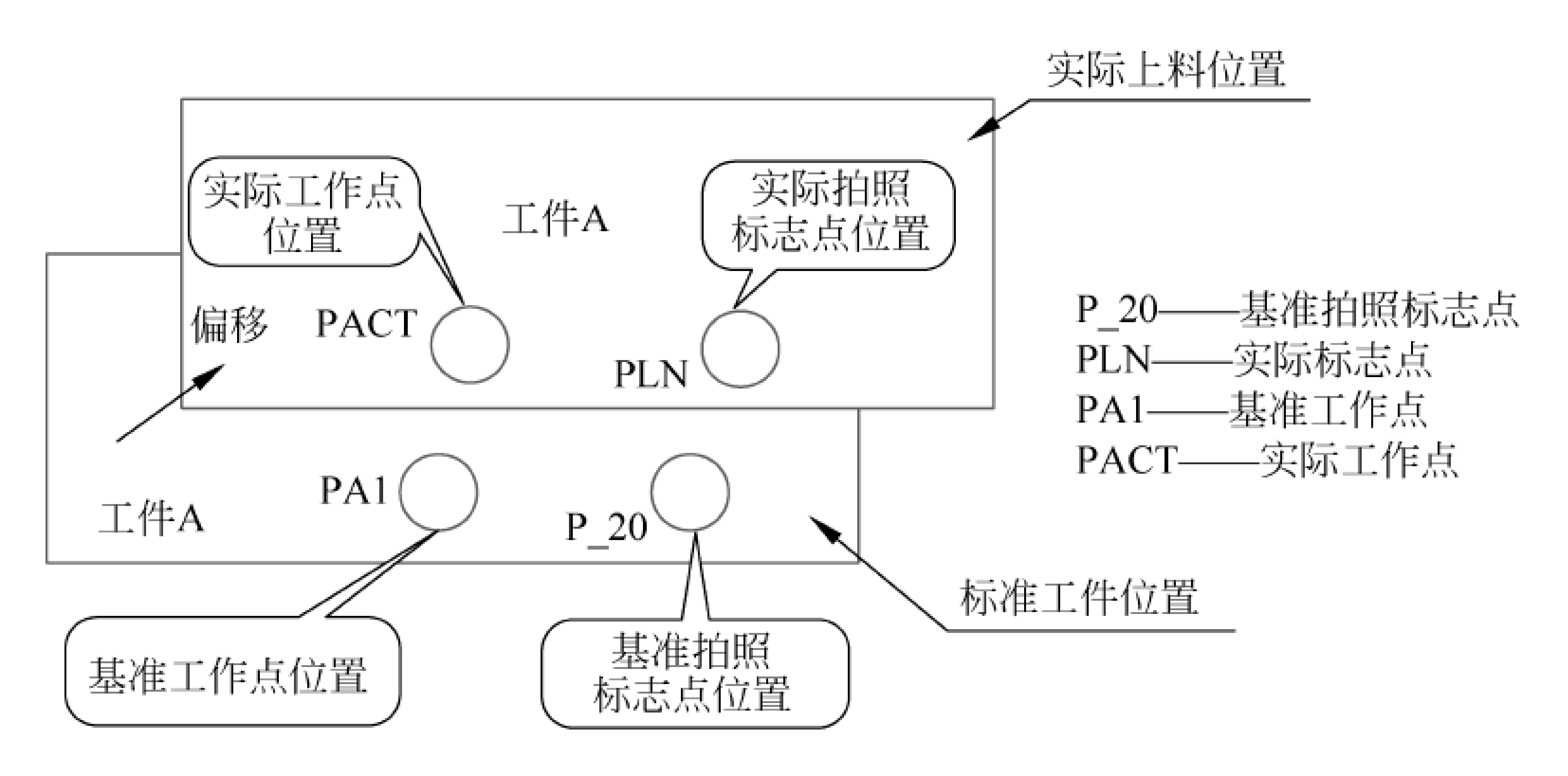


图 27-5　工况示意图

由于照相机安装在机器手上，所以拍照标志点与实际抓手工作点不是一个点。这是 P_20 与 PA1 的区别和不同用途。

如图 27-5 所示，每次上料后，工件 A 的基准标志点 P_20 偏移到 PLN 点。这样工件 B 的实际安装点就移动到 PACT 点。视觉照相机的功能就是要获得 PLN 点的坐标值。

2. 解决方案

(1) 预先测得基准标志点 P_20 坐标以及基准工作点 PACT 坐标，计算出这两点之间的关系（由于照相机安装在机器手上，所以拍照标志点与实际抓手工作点不是一个点）。

```
PA1 = P_20 * P_22
```

P_22——根据 P_20 点与 PA1 点相对位置求出。设置为全局变量，便于使用。

(2) 通过每次拍照获得实际标志 PLN 点的坐标值。

(3) 计算获得实际安装点坐标。

```
PACT = PLN * P_22
```

3. 操作步骤

(1) 确认工件上的基准拍照标志点。用示教单元获取该点的位置坐标，并命名为 P_20。

(2) 确认工件上的基准安装点。用示教单元获取该点的位置坐标，并命名为 PA1。

(3) 根据 PA1＝P_20 * P_22 公式，计算出 P_22。

(4) 移动机器人到拍照位，拍照并读取视觉信息，获得 PLN 点坐标值。

(5) 根据 PACT＝PLN * P_22 计算出实际安装点位置。

4. 机器人程序

```
'——判断位置回安全高度。
Servo On '——伺服 ON。
If P_Curr.Z > 850 Then '—— 判断当前位置点的 Z 轴坐标是否大于 850，如果大于则
Ovrd 20 '——设置速度倍率。
phome = P_Curr '——定义 phome 为当前位置。
phome.Z = 800 '——定义 phome 的高度为 800mm。
Mvs phome '——上升回 800m 高度。
EndIf'——判断语句结束。
GoTo *get1'——跳转到 *get1 行。
```

```
10 '---- 通信并接收数据 ---
11 *comm'——程序分支标志。
12 NVOpen "COM3:" As #1 '——将1#视觉传感器与通信口3建立通信通道。
13 Wait M_NvOpen(1) = 1 '——等待通信接口联机完成。
14 NVLoad #1,"AF1" '——加载程序 AF1。
15 NVRun #1,"AF1"'——启动相机的程序。AF1是视觉传感器用程序名。
16 DLY 0.5'——暂停0.5秒。
17 NVIn #1,"AF1","J21","K21","M21",0,10 '——读取位置型数据。
18 GoSub *data '——跳转子程序*data。
19 GoTo *putfs1'——跳转到*putfs1行。
20 Hlt  '——暂停,等待START再启动。
21 End  '——主程序结束。
22 *data'——数据整定。
23 PLN= P_NvS1(1) '——PLN为实际标志点视觉信息数据。
27 PACT= PLN * P_22. '——获得实际安装工作点位置数据(位置点乘法运算)。
36 Return  '——返回。
37 '——取抓手138 *get1。
39 Mov phand1get+(+0.00,+0.00,-200.00,+0.00,+0.00,+0.00) '——移动。
40 Ovrd 40  '——速度倍率40%(速度)。
41 Dly 0.5 '——暂停0.5秒。
42 Mvs phand1get+(+0.00,+0.00,-50.00,+0.00,+0.00,+0.00)  '——移动到抓取1号抓手上方50mm。
43 M_Out(14) = 1'——卡爪收回。
44 Ovrd 5 '——速度倍率5%。
45 Mvs phand1get '——降到抓取位置。
46 M_Out(14) = 0  '——打开卡爪。
47 Dly 0.5'——暂停0.5秒。
48 Mvs phand1get+(+0.00,+0.00,-50.00,+0.00,+0.00,+0.00)  '——回升到50mm上方。
49 Hlt  '——暂停
50 GoTo *getfs '——跳转到"*getfs"行。
51 '----- 抓取工件1。
52 *getfs'——程序分支标志。
53 Ovrd 40'——设置速度倍率。
54 Mvs phand1get+(+0.00,+0.00,-250.00,+0.00,+0.00,+0.00)  '——高速移动到抓手上方250mm高度。
55 Mov pfs1get+(+0.00,+0.00,-250.00,+0.00,+0.00,+0.00)  '——高速移动到取工件1上方250mm高度。
56 Mvs pfs1get+(+0.00,+0.00,-50.00,+0.00,+0.00,+0.00)  '——高速下降到工件1上方50mm。
57 M_Out(10) = 0  '——汽缸关闭。
58 M_Out(12) = 0'——输出12=0。
59 Ovrd 5'——设置速度倍率。
60 Mvs pfs1get '——低速下降到抓取工件1位置。
61 Dly 0.5'——暂停0.5秒。
62 M_Out(10) = 1  '——工件1夹取。
63 M_Out(12) = 1'——输出12=1。
64 Dly 0.5'——暂停0.5秒。
65 Mvs pfs1get+(+0.00,+0.00,-50.00,+0.00,+0.00,+0.00)  '——低速上升到工件1上方50mm。
66 Ovrd 40'——设置速度倍率。
67 Mvs pfs1get+(+0.00,+0.00,-250.00,+0.00,+0.00,+0.00)  '——高速上升到工件1上
```

方 250mm。
68 Hlt'——暂停。
69 GoTo * photo1'——跳转到" * photo1"行。
70 '——1 号工件 1 拍照。
71 * photo1'——程序分支标志。
73 Mov P_20.'——移动到标准拍照点。
74 HLT'——暂停。
75 GoTo * comm '—— 跳转到通信程序执行拍照。
76 '——安装工件 177 * putfs1。
78 HLT'——暂停。
81 Ovrd 5'——设置速度倍率。
82 Mvs PACT'——PACT 是实际的工件安装位置。
83 Dly 0.5'——暂停 0.5 秒。
END'——暂停。

27.3 需要思考的问题

(1) 在案例中,怎样获得偏差补偿量?
(2) 如何编制偏差的程序?
(3) 阐述解决问题的工作流程。
(4) 说明偏差量计算公式 PH=Inv(PVS0) * PVSCHK 的几何意义。
(5) 说明案例 2 的解决方案及操作流程。

第 28 章 第 28 日——视觉追踪功能的学习及应用

【学习目的】

视觉追踪功能是根据视觉系统的数据信息和传送带的运动信息驱动机器人动作的功能，因此牵涉三个方面，即机器人、视觉系统、传送带。在本章要学习视觉追踪系统的构成和机器人追踪功能的指令，对视觉追踪功能有一个框架性的了解。

28.1 概　　述

28.1.1 追踪功能

追踪功能指机器人追踪在传送带上的工件运动的功能，可以在传送带不停机的情况下抓取及搬运工件，不需要工件固定于某一位置。其特点如下。

(1) 能够追踪在传送带上线性整齐排列的工件并抓取搬运工件（使用光电开关检测在传送带上的工件位置）。

(2) 能够追踪在传送带上不规则排列的工件（包括不同种类的工件）并抓取搬运工件（使用视觉系统检测工件位置）。

(3) 追踪传送带运动速度变化的工件。

(4) 使用 MELFA-BASIC V 编程指令可以方便编制追踪程序。

(5) 使用采样程序可以方便构建系统。

28.1.2 一般应用案例

追踪功能一般应用于如下场合。

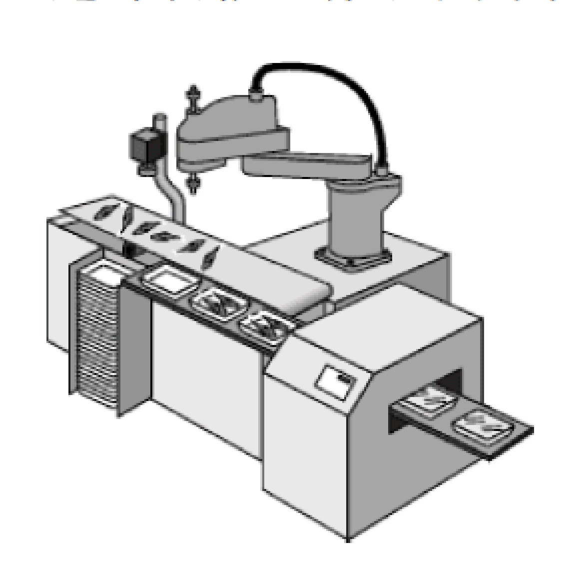


图 28-1　在食品生产流水线上使用机器人进行追踪

1. 食品加工流水线

图 28-1 表示在食品生产流水线上使用机器人进行追踪抓取摆放成品。

2. 将工件整齐排列

图 28-2 是将生产流水线上随机凌乱排列的工件摆放整齐的案例。

3. 小型电子产品的装配

图 28-3 是使用机器人的追踪功能进行小型电子产品装配的案例。

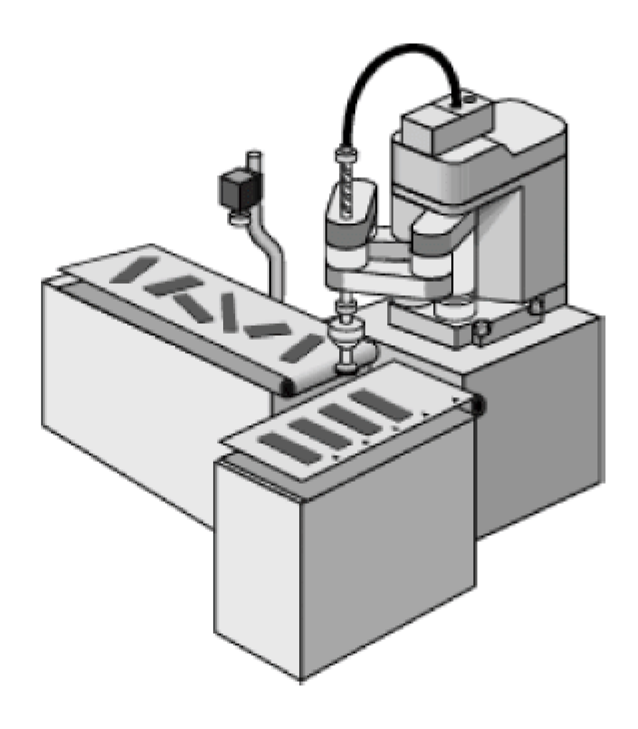


图 28-2　将生产流水线上随机排列的工件摆放整齐

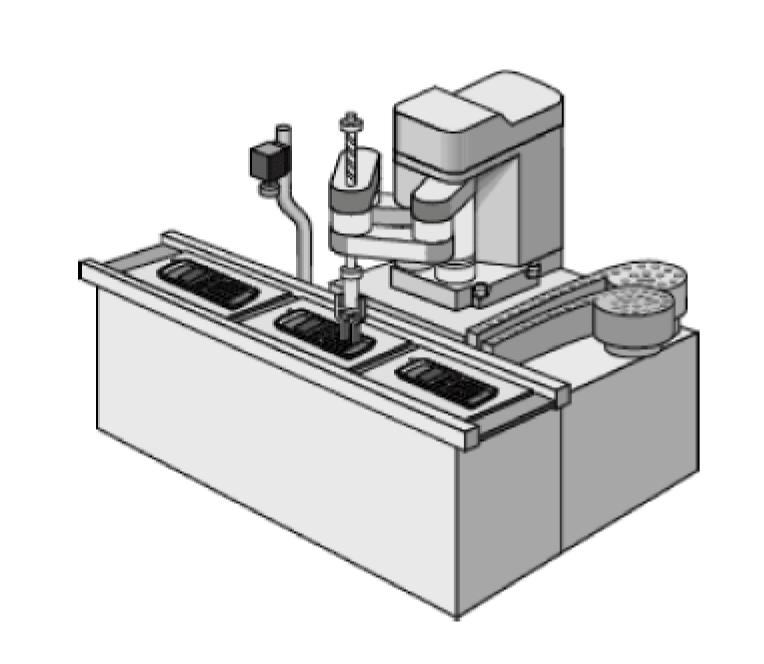


图 28-3　使用机器人的追踪功能进行小型电子产品装配

28.1.3　追踪功能技术术语和缩写

关于追踪功能的部分技术术语及缩写如表 28-1 所示。

表 28-1　追踪功能的部分名称及缩写

名　　词	功　　能
追踪功能	追踪功能即机器人追踪在传送带上的工件运动，在传送带不停机的情况下抓取及搬运工件的功能
传送带追踪	传送带上的工件为线性整齐排列时，机器人根据光电开关的信息进行追踪工作的模式
视觉追踪	当传送带上的工件为不规则排列时，机器人根据视觉系统提供的信息进行追踪的工作模式
网络视觉传感器	用于识别工件的视觉传感器系统
编码器编号	由参数 EXTENC 设置的编码器编号。表示在追踪功能中使用的编码器序号
TREN signal	追踪功能使能信号

28.1.4　可构成的追踪应用系统

除了在 28.1.2 节列举的追踪功能应用场合外，根据机器人控制器的功能，还可以进行如表 28-2 所示的应用。

表 28-2　可构成的追踪应用系统

序　　号	控制器 CR750-D CR751-D	系 统 样 例
1	OK	机器人抓取在传送带上运行的工件
2	OK	从托盘上抓取工件放置在传送带上
3	OK	将工件放置在机器人上方的 S 形挂钩上
4	OK	在追踪过程中，机器人对传送带上的工件进行加工处理
5	OK	在追踪过程中，对传送带上的工件进行装配
6	OK	能够对传送带 A 和传送带 B 进行追踪
7	OK	能够使用差动型编码器进行追踪
8	OK	能够使用电压型编码器和集电极开路型编码器构成追踪系统

28.2 硬件系统构成

本节叙述构成追踪系统所需要的最基本的硬件，即除了机器人本体外所需要的硬件。

28.2.1 传送带追踪用部件构成

传送带追踪系统所需要的硬件如表 28-3 所示。

表 28-3 传送带追踪系统所需要的硬件

部件名称	型号	数量	说明
机器人部分			
示教单元	R32TB/R33TB	1	
抓手		1	
抓手传感器		1	用于确认抓手抓牢工件
电磁阀套件		1	
抓手输入电缆		1	
气阀输入接口	2A-RZ365 或 2A-RZ375	1	
标定用检具		1	
传送带部分			
编码器		N	推荐使用编码器型号 OmronE6B2-CWZ1X-1000/－2000 编码器电缆为带屏蔽双绞线
光电开关		N	用于同步追踪系统
5V 电源		N	用于编码器
24V 电源		N	用于光电开关
个人计算机			
个人计算机			
RT ToolBox2	3D-11C-WINE 3D-12C-WINE		

28.2.2 视觉追踪系统部件构成

视觉追踪系统所需要的硬件如表 28-4 所示。

表 28-4 视觉追踪系统所需要的硬件

部件名称	型号	数量	备注
机器人部分			
示教单元	R32TB/R33TB	1	
抓手		1	
抓手传感器		1	用于确认抓手抓牢工件
电磁阀套件		1	
抓手输入电缆		1	
气阀输入接口	2A-RZ365 或 2A-RZ375	1	

续表

部件名称	型　　号	数　　量	备　　注
标定用检具		1	
传送带部分			
编码器		N	推荐使用编码器型号 OmronE6B2-CWZ1X-1000/—2000 编码器电缆为带屏蔽双绞线
光电开关		N	用于同步追踪系统
5V 电源		N	用于编码器
24V 电源		N	用于光电开关
视觉系统			
视觉系统	4D-2CG5xxxx-PKG	1 套	
镜头		1	
照明设备		1	
连接部件			
Hub		1	
以太网电缆		1	机器人控制器与 Hub 之间连接；计算机与 Hub 之间连接
个人计算机			
个人计算机			
RT ToolBox2	3D-11C-WINE 3D-12C-WINE		

28.2.3　传送带追踪系统构成案例

本节以图示方式说明追踪系统的构成和布置，如图 28-4 和图 28-5 所示。

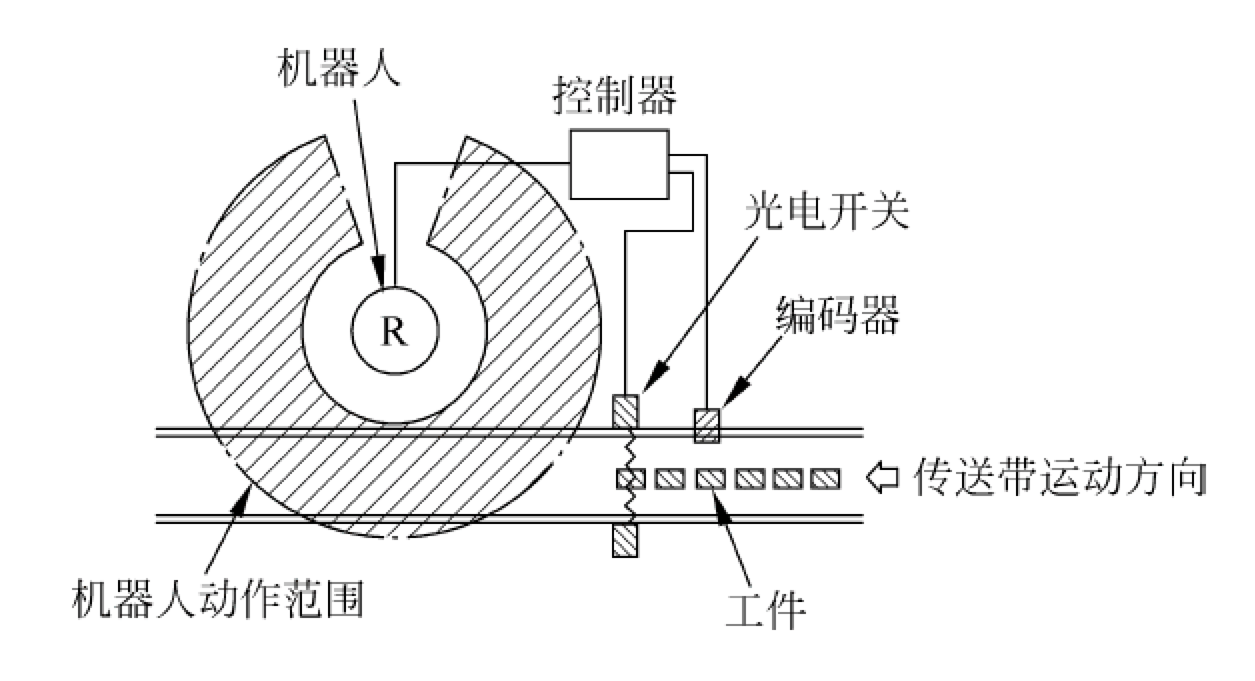


图 28-4　传送带追踪系统的构成和布置

在传送带追踪系统中，机器人的动作范围必须覆盖传送带运行区域的一部分。传送带上的工件是线性整齐排列的(即在机器人坐标系中，工件位置的 X 和 Z 坐标都不变化，只有 Y 坐标不断变化)，而且：

(1) 使用简单的光电开关检测工件经过的位置，光电开关的检测信号输入到机器人控制器的通用 I/O 单元。光电开关使用 DC24V 电源。

(2) 使用编码器检测传送带的运动速度，也间接表示了在传送带上工件的运动位置。

编码器信号直接输入到机器人控制器中。编码器使用 DC5V 电源。

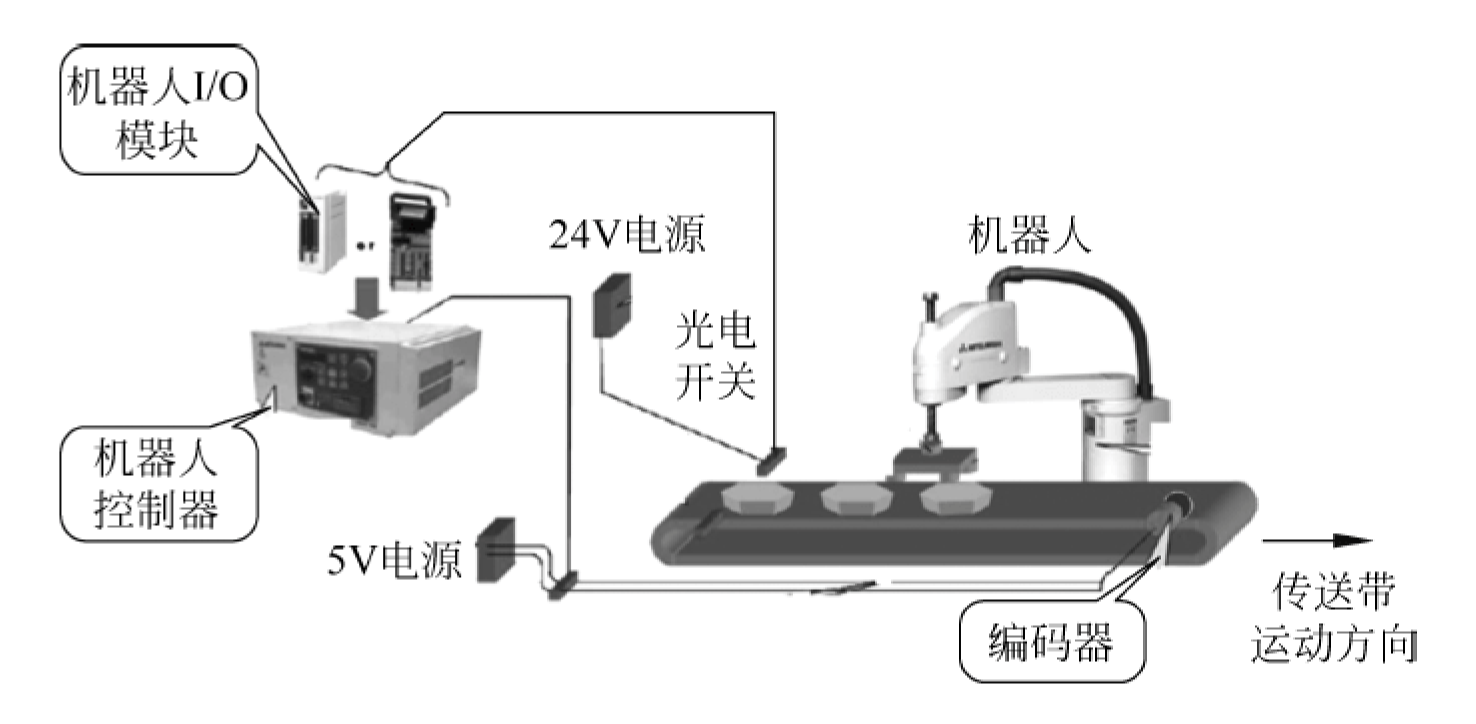


图 28-5　传送带追踪系统的实物构成图

28.2.4　视觉追踪系统构成案例

本节以图示方式说明视觉追踪系统的构成和布置，如图 28-6 和图 28-7 所示。

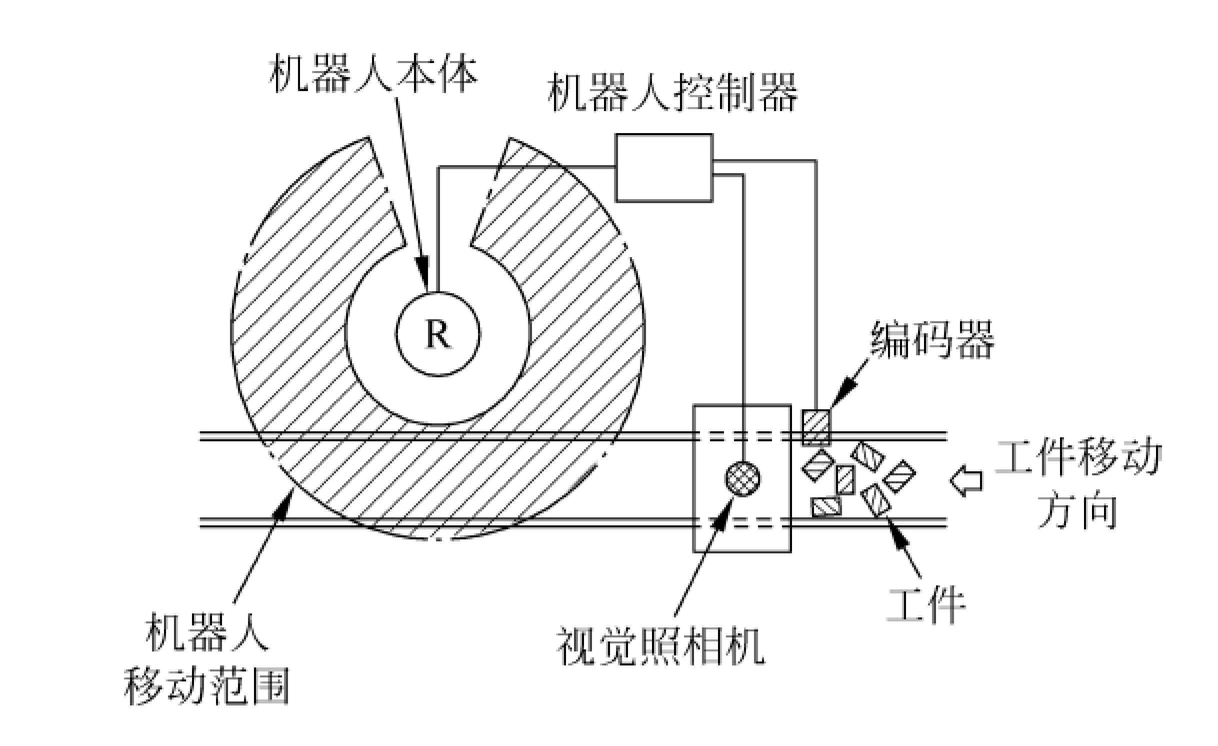


图 28-6　视觉追踪系统的构成和布置

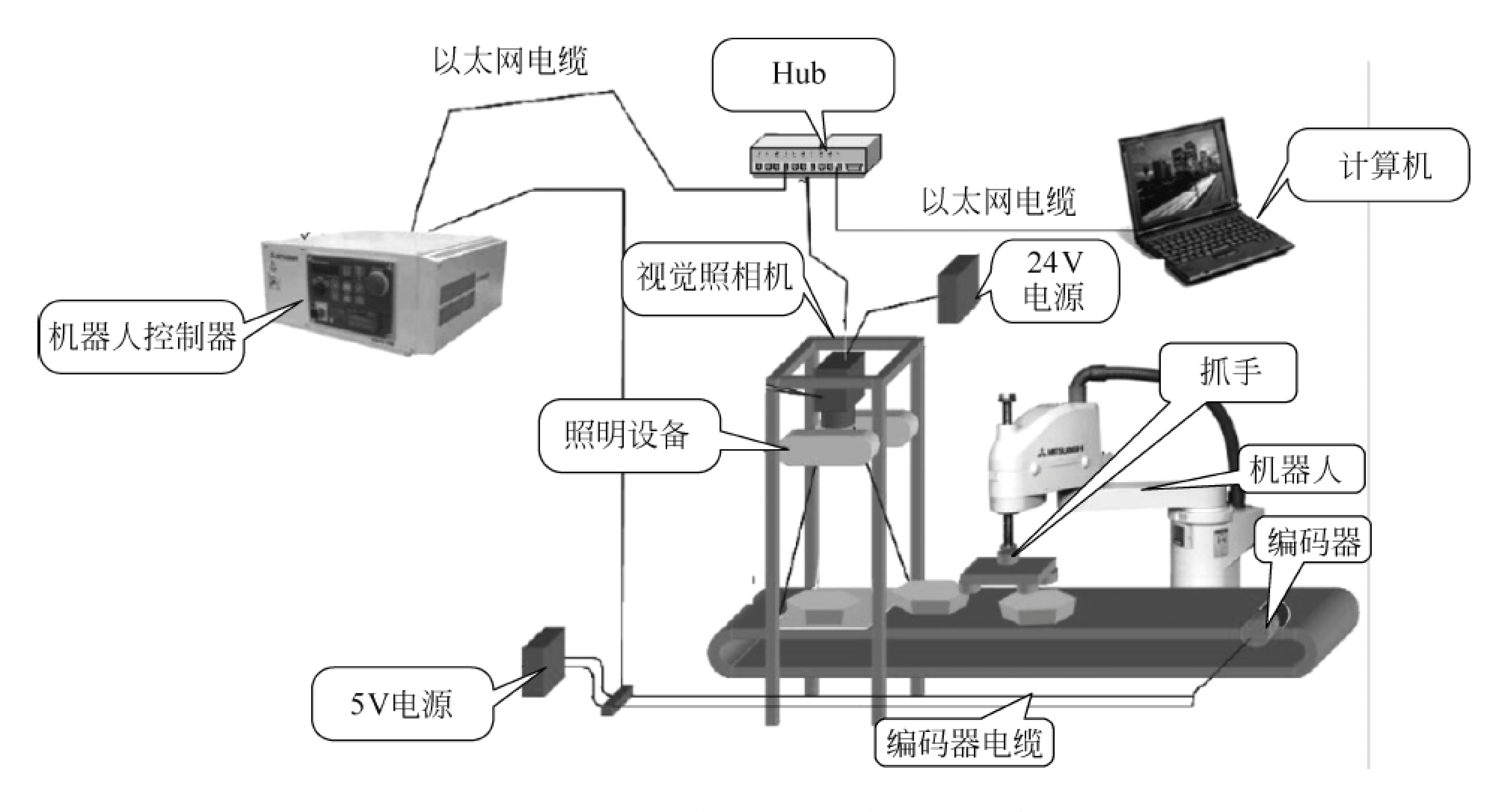


图 28-7　视觉追踪系统的实物构成和布置

在视觉追踪系统中，机器人的动作范围必须覆盖传送带运行区域的一部分。传送带上的工件是无规则排列的（即在机器人坐标系中，工件位置的 X、Y、Z 坐标都可能变化），所以：

（1）使用视觉系统检测工件的位置，视觉系统通过以太网与机器人控制通信，将检测识别到的工件位置信息传送到机器人控制器。视觉系统使用 DC24V 电源。

（2）使用编码器检测传送带的运动速度，也间接表示了在传送带上工件的运动位置。编码器信号直接输入到机器人控制器中。编码器使用 DC5V 电源。

28.3 技术规格

在构成追踪系统时，首先会确定追踪系统中传送带的速度和抓取位置精度，这就是追踪系统的部分技术规格。

如表 28-5 所示是追踪功能技术规范。

表 28-5 追踪功能技术规范

项目		规范
适用机器人		RV-3SD/6SD/12SD 系列 RH-6SDH/12SDH/18SDH 系列 RH-FH-D 系列
适用控制器		CR1D/CR2D/CR3D CR750-D/CR751-D
机器人程序语言		配置有追踪功能的机器人语言
传送带	运动速度	300mm/s（机器人频繁运送工件） 500mm/s（工件间隔较大） 一个机器人可服务于两条传送带
	编码器	输出格式：A，A/，B，B/，Z，Z/ 输出规范：线性驱动 最高响应频率：100kHz 分辨率：最高 2000 脉冲/转 推荐型号：Omron E6B2-CWZ1X-1000 E6B2-CWZ1X-2000
	编码器电缆	24AWG（0.2mm^2） 电缆长度：最长 25m。屏蔽双绞线
光电开关		
视觉系统		
位置精度		约±2mm（传送带速度为 300mm/s 时）

28.4 追踪工作流程

本节叙述追踪工作的流程，有些工作将在以下的章节中介绍，本节仅起到一个提纲挈领的作用，按照如图 28-8 所示的工作流程就不会发生紊乱。

对工作流程的说明如下。

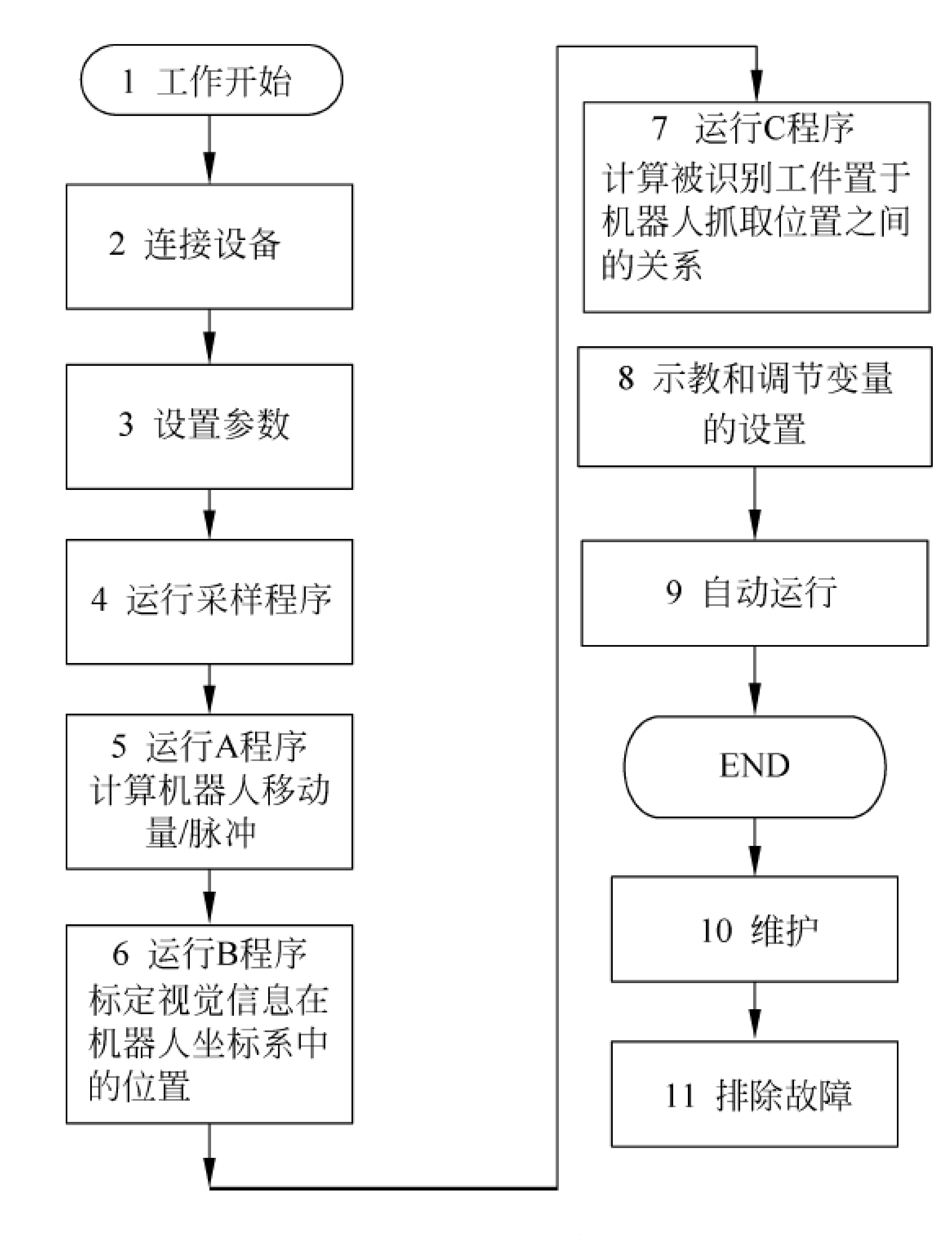


图 28-8 追踪工作流程图

(1) 工作开始。

(2) 连接设备——机器人本体连接安装及初始化。

① 机器人本体安装及自带电缆的连接;

② 机器人原点设置及初始化设置;

③ 机器人 I/O 信号端子排制作及安装连接;

④ 机器人操作面板的制作及连接;

⑤ 编码器端子头的制作及电缆制作连接;

⑥ 光电开关安装及信号连接;

⑦ 视觉系统安装以及以太网连接;

⑧ 编码器安装及电缆连接。

(3) 设置参数。

根据 20.6 节的说明进行参数设置,参数是必不可少的。

(4) 进行各信号的有效性检查。

① 检查传送带运行编码器数值是否有变化。

② 检查光电开关信号是否有效。

③ 检查是否能够接收到视觉系统发出的数据。

(5) 运行 A 程序:进行单位脉冲机器人移动量标定。

(6) 运行 B 程序:进行视觉坐标系与机器人坐标系关系的标定。

(7) 运行 C 程序:标定"抓取点""待避点""下料摆放点"。

(8) 试运行 1#程序及 CM1 程序,调整抓取点位置补偿量。

(9) 维护。

(10) 结束。

28.5 设备连接

28.5.1 编码器电缆的连接

一台机器人控制器最多连接两台编码器,使用 E6B-2-CWZ1X(OMRON 编码器)。如图 28-9 所示,编码器共有 8 根信号线需要连接。

(1) 将编码器厂家提供的编码器电缆连接在中继端子排上;

(2) 通过中继端子排与机器人厂家提供的插头相连;

(3) 将插头插入机器人控制器;

(4) 特别注意必须将屏蔽线接地。

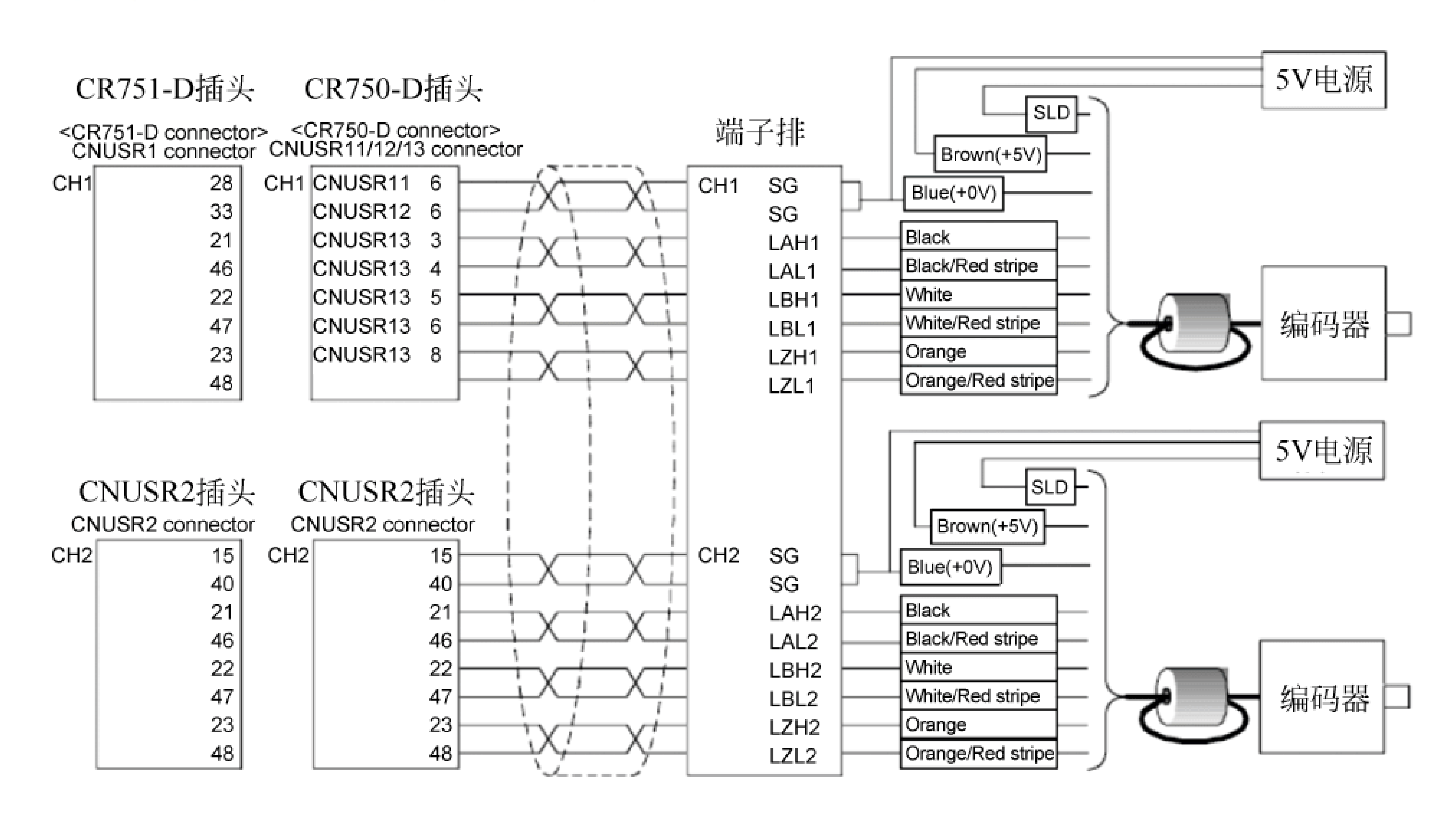


图 28-9 编码器信号电缆连接图

28.5.2 编码器电缆与控制器的连接

编码器电缆最终必须连接到机器人控制器上。控制器上有两个插口,不同的插口对应的"编码器编号"不同。"编码器编号"是个重要的概念,在后续的编程中会常常提到,所以在连接时必须特别注意。不同控制器其编码器插口位置不一样,请注意如图 28-10 及图 28-11 所示的位置。

28.5.3 抗干扰措施

在现场使用机器人追踪系统时,电磁干扰可能会对编码器信号造成干扰,有时甚至是极大的干扰,造成机器人不能够正常地工作甚至是误动作,所以必须采取基本的抗干扰措施。如图 28-12 所示,至少必须采取以下三项措施。

图 28-10　CR750-D 控制器与编码器电缆连接图

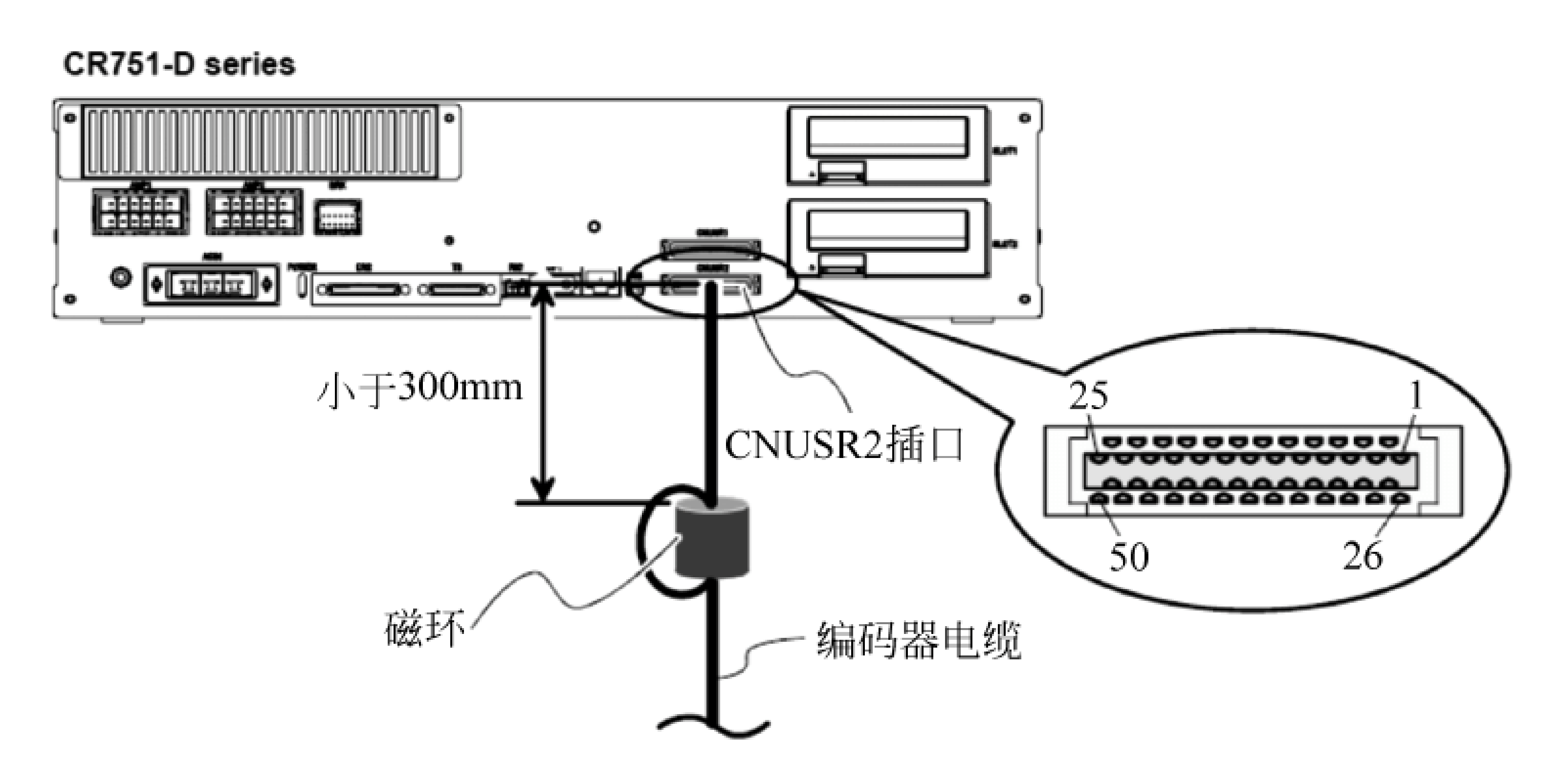


图 28-11　CR751-D 控制器与编码器电缆连接图

(1) 在 AC 电源侧加装线性滤波器；

(2) 在电缆上加装磁环；

(3) 必须保证良好的接地。接地线应该大于 14mm^2。

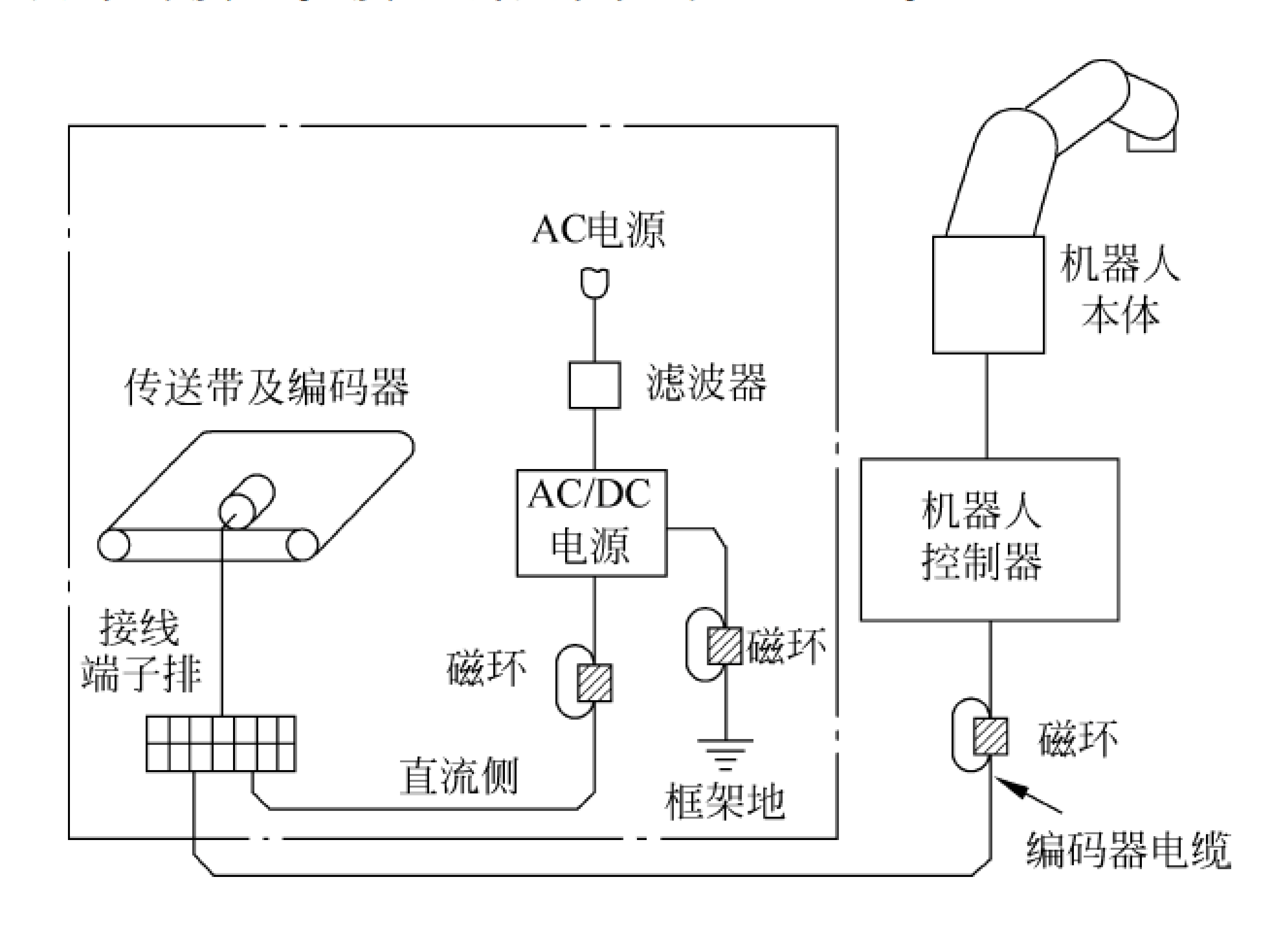


图 28-12　抗干扰措施

28.5.4 与光电开关的连接

光电开关的信号直接作为输入信号接入机器人控制器的通用 I/O 单元中。注意源型接法和漏型接法有所不同。

图 28-13 是光电开关连接示意图。光电开关一般使用 DC24V 电源。图 28-14 是源型接法的连接图。

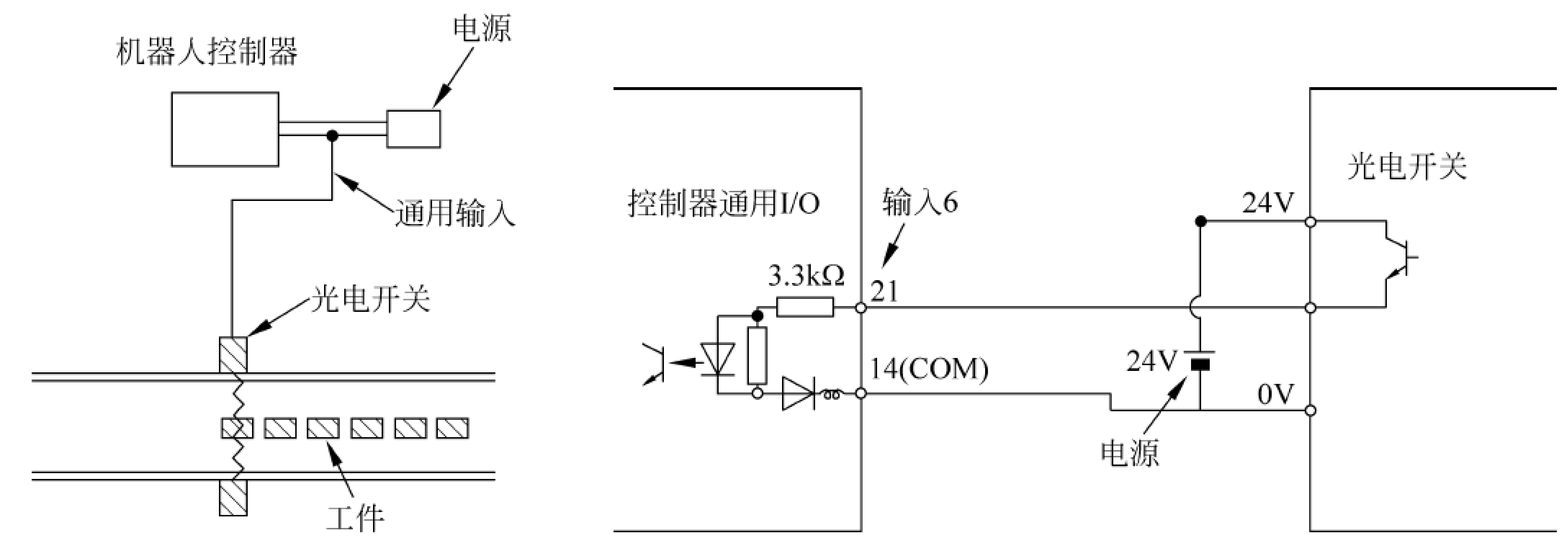


图 28-13 光电开关与控制器的连接　　图 28-14 光电开关与控制器源型接法示意图

28.6 参数的定义及设置

本节解释有关追踪参数的定义和设置，如表 28-6 所示。

表 28-6 追踪参数

<table>
<tr><th>参　　数</th><th>参　数　名</th><th></th><th>功 能 说 明</th><th>出　厂　值</th></tr>
<tr><td>追踪模式</td><td>TRMODE</td><td></td><td>追踪功能使能
0：不能　　1：使能</td><td>0</td></tr>
<tr><td colspan="5">参数的编辑
参数名：TRMODE　　机器号：0
说明：tracking permission[0:disable 1:enable]
1：1</td></tr>
<tr><td>编码器连接通道编号</td><td>EXTENC</td><td></td><td>设置编码器连接通道编号。指明编码器连接在控制器的哪一个通道口上
<table>
<tr><th>连接通道</th><th>编码器编号</th><th>备注</th></tr>
<tr><td>标准通道 1</td><td>1</td><td></td></tr>
<tr><td>标准通道 2</td><td>2</td><td></td></tr>
<tr><td>Slot1</td><td>CH1</td><td rowspan="4">后续扩展用</td></tr>
<tr><td>Slot1</td><td>CH2</td></tr>
<tr><td>Slot2</td><td>CH1</td></tr>
<tr><td>Slot2</td><td>CH2</td></tr>
<tr><td>Slot3</td><td>CH1</td><td rowspan="2"></td></tr>
<tr><td>Slot3</td><td>CH2</td></tr>
</table></td><td></td></tr>
</table>

续表

参　　数	参　数　名		功 能 说 明	出　厂　值
参数的编辑 参数名：EXTENC　机器号：0 说明：No. of external encodor 1：1　5：1 2：2　6：2 3：1　7：1 4：2　8：2				
工件间隔(判断)距离	TRCWDST		工件的间隔距离。当工件通过传感器时,传感器可能多次发出信号。机器人控制器可能将一个工件视为两个或更多工件,因此设置这个参数,使传感器在两个工件之间不发信号	5
参数的编辑 参数名：TRCWDST　机器号：0 说明：Work check distance for tracking[mm] 1：5.00				

28.7　追踪功能指令及状态变量

28.7.1　追踪功能指令及状态变量一览

1. 指令一览表

如表28-7所示为指令一览表。

表28-7　指令一览表

指 令 名 称	功　　能
TrBase	设置追踪模式的“追踪起点”和“编码器编号”
TrClr	对“追踪缓存区”清零
Trk	启动或结束“追踪模式”。
TrOut	输出信号并读编码器数据
TrRd	从“追踪缓存区”读工件数据
TrWrt	向“追踪数据缓存区”写工件数据

2. 状态变量一览表

如表28-8所示为状态变量一览表。

表 28-8　状态变量一览表

变量名称	功　能	属性	数据类型
M_Enc	编码器数值	R/W	双精度实数
M_EncL	已经存储的编码器数值	R/W	双精度实数
P_EncDlt	编码器每一脉冲机器人移动量。这一数据由程序 A 生成	R/W	位置数据
M_Trbfct	存储在“追踪数据缓存区”的数据数量	R	整数
P_Cvspd	传送带速度(mm，rad/s)	R	位置数据
M_EncMax	编码器数据最大值	R	双精度实数
M_EncMin	编码器数据最小值	R	双精度实数
M_EncSpd	编码器速度(脉冲/秒)	R	单精度实数
M_TrkCQ	追踪操作状态 1:追踪模态 0:非追踪模态	R	实数

3. 相关函数功能一览表

如表 28-9 所示为相关函数功能一览表。

表 28-9　相关函数功能一览表

函 数 名 称	功　能	数 据 类 型
Poscq(< position >)	检查指定的点位是否在设置范围内	整数
TrWcur(< encoder number >, < position >,< encoder value >)	获得当前工件位置	位置
TrPos(< position >)	获取在追踪起点的工件位置	位置

28.7.2　追踪功能指令说明

1. TrBase 追踪基本指令

1）功能

本指令用于设定“追踪原点”和使用的“编码器编号”。

2）格式

```
TrBase  <原点位置> [ , <编码器编号>]
```

3）术语说明

(1) <原点位置>——追踪运行中的“追踪的起点”位置。

(2) <编码器编号>——使用的编码器连接到控制器的通道号。

4）样例

```
1  TrBase P0 '—— 以 P0 为"追踪原点"。
2  TrRd P1,M1,MKIND '——从追踪缓存区读出"被检测到的数据"。
3  Trk On,P1,M1 '——以 P1 和 M1 为对象进行追踪。
4  Mvs P2 '——设置 P1 的当前位置为 P1c,使得机器人跟踪工件操作中的目标位置为 "P1c * P_
Zero/P0 * P2"。
```

这说明追踪过程中目标位置一直在改变(事实上也一直在变化)。系统能够识别的目标

位置="P1c * P_Zero/P0 * P2"。

P2是实际程序中的抓取点。

```
5  HClose 1 '——抓手闭合。
6  Trk Off '——停止追踪。
```

5）解释

本指令用于设置"追踪原点"以及"编码器编号"。如果没有标写编码器编号则为预置值"1"。在控制器中设置有"追踪原点"及"编码器编号"的初始值，使用 TrBase 或 Trk 指令可以改变初始值。

"追踪原点"初始值= P_Zero，"编码器编号"初始值=1。

2. TrClr 追踪缓存区数据清零指令

1）功能

清除追踪缓存区中的数据。

2）格式

```
TrClr [<缓存区编号>]
```

3）术语

<缓存区序号>：追踪数据缓存区的序号。设置范围为1～4。

4）样例

```
1  TrClr 1 '——清除1#追踪缓存区内的数据。
2   * LOOP'——程序分支标志。
3  If M_In(8) = 0 Then GoTo  * LOOP '——判断语句。
4  M1# = M_Enc(1) '——将当前编码器数值代入 M1#。
5  TrWrt P1, M1#,MK '——写入数据。
```

5）说明

（1）清除存储在追踪缓存区内的数据。

（2）在追踪程序初始化时使用本指令。

3. Trk 追踪功能指令

1）功能

Trk On——机器人进入追踪模式工作。

Trk Off ——停止追踪。

2）格式

```
Trk On[,<测量位置数据>[,[<编码器数值>][,[<追踪原点>][,[<编码器编号>] ] ] ] ]
Trk Off
```

3）术语

（1）<测量位置数据>（可省略）：设置由传感器检测到的工件位置。

（2）<编码器数据>（可省略）：设置当检测到工件时编码器的数值。

（3）<追踪原点位置>（可省略）：设置追踪模式中使用的原点。如果省略，则使用 TrBase 指令设置的原点。其初始值= PZERO。

（4）<编码器编号>（可省略）。

4）样例

```
TrBase P0 '——设置 P0 为追踪原点。
TrRd P1,M1,MKIND '——读出"追踪缓存区"数据。
Trk On,P1,M1 '—— 追踪启动。以位置数据 P1、编码器数据 M1 为基准进行追踪。
Mvs P2 '—— 移动到 P2 点。
HClose 1 '——抓手 1 = ON。
Trk Off '——追踪模式结束。
```

5）说明

追踪工作以检测点的“工件位置”和检测点动作时的“编码器数值”为基础进行追踪。

4. TrOut 输出信号和读取编码器数值指令

1）功能

指定一个输出信号＝ON 和读取编码器数值。

2）格式

```
TrOut <输出信号编号(地址)>, <存储编码器 1 数值的变量> [ , [<存储编码器 2 数值的变量]
[ , [<存储编码器 3 数值的变量>] [ , [<存储编码器 4 数值的变量>]
[ , [<存储编码器 5 数值的变量>] [ , [<存储编码器 6 数值的变量]
[ , [<存储编码器 7 数值的变量>] [ , [<存储编码器 8 数值的变量>] ]]]]]]]
```

3）样例

```
 * LOOP1   '——程序分支标志
If M_In(10) <> 1 GoTo * LOOP1 '——检查外部信号(光电开关)是否动作。没有动作就继续等待。
(如果光电开关 = ON,就执行下一步。)
TrOut 20, M1 # , M2 # '——指定输出信号 20 = ON,同时指定编码器 1 的数据存储在 M1 # 变量,编
码器 2 的数据存储在 M2 # 变量。存储过程与输出信号 = ON 同步。
 * LOOP2 '——程序分支标志。
If M_In(21) <> 1 GoTo * LOOP2 '——如果 M_In(21) = 1,则
M_Out(20) = 0 '——设置 M_Out(20) = 0。
```

4）说明

(1) 本指令指定一个输出信号＝ON 和读取编码器数值。

(2) 本指令可以在输出计算指令的同时获取由编码器测量的工件位置信息。

(3) 因为通用输出信号是可以保持的，因此，在确认已经获取视觉系统信息后，必须使用 M_Out 变量使通用输出信号＝OFF。

5. TrRd 读追踪数据指令

1）功能

从“追踪数据缓存区”中读位置追踪数据以及编码器数据等。

2）格式

```
TrRd <位置数据> [ , <编码器数据>] [ , [<工件类型数>] [ , [<缓存区序号>] [ , <编码器编号>] ] ]
]
```

3）术语

(1) <位置数据>(不能省略)：设置一个变量用于存储从缓存区中读出的工件位置。

(2) <编码器数值>(可省略)：设置一个变量用于存储从缓存区中读出的编码器数值。

(3) <工件类型数量>(可省略)：设置一个变量用于存储从缓存区中读出的工件类型数。

(4) <缓存区序号>(可省略)：设置被读出数据的缓存区序号。如果设置为1则可省略。设置范围为1～4。与参数TRBUF相关

(5) <编码器编号>(可省略)：设置一个变量用于存储从缓存区中读出的编码器编号。

4) 样例

(1) 追踪操作程序

```
1  TrBase P0 '——设置 P0 为追踪原点。
2  TrRd P1,M1,MK '——读数据指令。读出的位置数据存储在 P1,编码器数据存储在 M1,工件类型数量存储在 MK。
3  Trk On,P1,M1 '——追踪启动。工件检测点 = P1,同时编码器数据为 M1。
4  Mvs P2   '——前进到 P2 点。
5  HClose 1 '——抓手闭合。
6  Trk Off '——追踪操作结束。
```

(2) 传感器数据接收程序

```
1  * LOOP'——程序分支标志。
2  If M_In(8) = 0 Then GoTo * LOOP '——M_In(8)是光电开关检测信号。
3  M1 # = M_Enc(1) '——将当前编码器数值代入 M1 #。
4  TrRd P1,M1,MK——读数据指令。读出的位置数据存储在 P1,编码器数据存储在 M1,工件类型数量存储在 MK。
```

5) 说明

(1) 本指令读出由TrWrt指令写入指定缓存区的各数据：工件位置、编码器数值、工件类型数等。

(2) 如果执行本指令时,在指定的缓存区内没有数据,则发出报警。报警号2540。

6. TrWrt 写追踪数据指令

1) 功能

在追踪操作中,将位置数据、编码器数值写入“追踪数据缓存区”中。

2) 格式

```
TrWrt <位置数据> [ , <编码器数值>] [ , [<工件类型数>] [ , [<缓存区序号>] [ , <编码器编号>] ] ] ]
```

3) 术语

(1) <位置数据>(不能省略)：指定由传感器检测的位置数据。

(2) <编码器数值>(可省略)：指在工件被检测到的位置点的编码器数值。获取的编码器数值存储在M_Enc()状态变量中并通常由TrOut指令指定。

(3) <工件类型数量>(可省略)：指定工件类型数量。设置范围为1～65 535。

(4) <缓存区序号>(可省略)：指定数据缓存区序号。设置=1时,可以省略。设置范围为1～4。

(5) <编码器编号>(可省略)：设置外部编码器编号。

4) 样例

(1) 追踪操作程序

```
1  TrBase P0'——设置 P0 为追踪原点。
```

```
2  TrRd P1,M1,MKIND '——读数据指令。
3  Trk On,P1,M1 '——追踪启动。
4  Mvs P2   '——前进到 P2 点。
5  HClose 1 '——抓手闭合。
6  Trk Off   '——追踪结束。
```

(2) 光电传感器程序

```
1   *LOOP '——程序分支标志。
2  If M_In(8) = 0 Then GoTo *LOOP '——如果光电开关的输入信号 = OFF,就反复循环等待。否则
3  M1# = M_Enc(1) '——如果光电开关输入信号 8 = ON,就将此时的编码器数据赋予 M1#。
4  TrWrt P1, M1#,MK '—— 同时将此点位的工件数据 P1,编码器数据 M1# 和工件类型数写入缓
存区。
```

5) 说明

(1) 本指令将测量点(检测开关=ON)的工件位置数据,编码器数值、工件类型数和编码器编号写入“追踪数据缓存区”。

(2) 除工件位置外的其他参数可省略。

(3) 用参数 TRCWDST 设置工件间隔距离,如果工件在间隔之内,就被视为同一工件。即使数据被写入两次或两次以上,也只有一个数据被存储在缓存区。因此使用 TrRd 指令只会读出一个数据。

28.8 故障排除

28.8.1 报警号在 9100～9900 的故障

表 28-10 为报警 9100～9900 故障一览表。

表 28-10 报警 9100～9900 故障一览表

报警编号	故障现象	故障原因及排除方法
9100	通信故障	原因:在 C 程序中视觉系统与机器人未正确连接或机器人未能登录在视觉系统中。 处理:检查连接机器人和视觉系统的以太网电缆
9101	编码器编号超范围	原因:编码器编号设置超出范围。 处理:检查程序中 PE 变量的 X 坐标值
9102	工件类型数超范围	原因:设置的“工件类型数”超出范围。 处理:检查“C 程序”中变量 PRM1 的 X 坐标值。 如果该值大于 11,则要修改 C 程序中的程序行“MWKMAX=10”
9110	位置精度超范围	原因:由 A 程序和 C 程序计算的位置精度与理论值相差很大。 处理: (1) 检查 CM1 程序中的位置变量 PVTR 的 X 和 Y 坐标值。这些数值表示了与理论值的差别。 (2) 如果这些数值相差过大,则再次运行“A 程序”“B 程序”“C 程序”。 (3) 检查“CM1 程序”中的位置变量 PCHK 的 X 和 Y 坐标值是否不为“0”。如果=0,则需要修改这些差值到允许的精度

续表

报警编号	故障现象	故障原因及排除方法
9199	程序错误	原因：在 1# 程序中的 * S50WKPOS 程序不能生成一个返回值。 处理：检查 * S50WKPOS 程序中的 MY50STS 值不能从 0 改变为其他值的原因

28.8.2 其他报警

表 28-11 为其他故障报警一览表。

表 28-11 其他故障报警一览表

报警号	故障现象	故障原因及排除方法
L2500	编码器数据错误	原因：编码器数值不正常 处理： (1) 以固定速度检查传送带的旋转状态； (2) 检查编码器连接状态； (3) 检查地线及接地
L2510	追踪参数错误	原因：追踪参数 EXCRGMN 和 EXCRGMX 设置值相反 处理：检查 ENCRGMX 和 ENCRGMN 的设置值
L2520	追踪参数超范围	原因：参数 TRBUF 的设置值超范围。 处理：检查并重新设置参数 TRBUF。
L2530		原因：缓存区数据写入错误。 处理： (1) 检查 TrWrt 指令的执行次数是否正确。 (2) 检查参数 TRBUF 的设置是否正确。 (3) 检查“CM1 程序”中的位置变量 PCHK 的 X 和 Y 坐标值是否不为 0。如果为 0，则需要修改这些差值到允许的精度
L2540	没有数据被写进缓存区，所以读不出数据	原因：没有数据被写进缓存区，所以读不出数据。 处理： (1) 使用状态变量 M_Trbfct 确认缓存区内有数据之后再执行 TrRd 指令。 (2) 确认读指令与写指令的缓存区序号相同
L2560	追踪参数不当	原因：EXTENC 参数设置超出范围。设置范围 1～8
L3982	点位不当不能使用	原因：奇异点
L6632	不能写入 TREN 信号	原因：在实时信号输入模式中，外部输出信号 810～817 不能被写入。 处理：使用实时输入信号 TREN

28.9 参数汇总

与追踪功能相关的参数如表 28-12 所示。

表 28-12　与追踪功能相关的参数

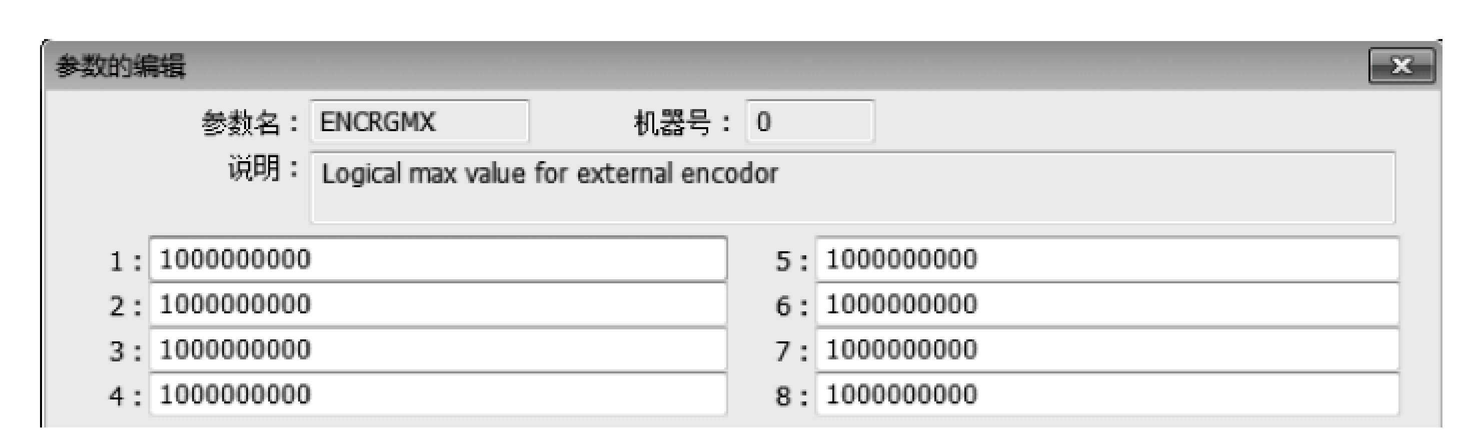
参数名称	参数简写	功　　能	出　厂　值
追踪缓存区	TRBUF	追踪缓存区的编号及大小(KB) 追踪缓存区用于存储追踪工作的数据。主要存储每一传送带的数据。当增加传送带时就要改变设置值。设置范围 1～8。缓存区大小设置范围 1～200KB	2,64

参数的编辑
参数名：TRBUF　机器号：0
说明：Tracking data number of buffer, size of buffer
1：2
2：64

参数名称	参数简写	功　　能	出　厂　值
编码器数据最小值	ENCRGMN	编码器数据最小值 编码器数值由状态变量 M_Enc 获得	0,0,0,0,0,0,0,0



参数名称	参数简写	功　　能	出　厂　值
编码器数据最大值	ENCRGMX	编码器数据最大值 编码器数值由状态变量 M_Enc 获得	100000000

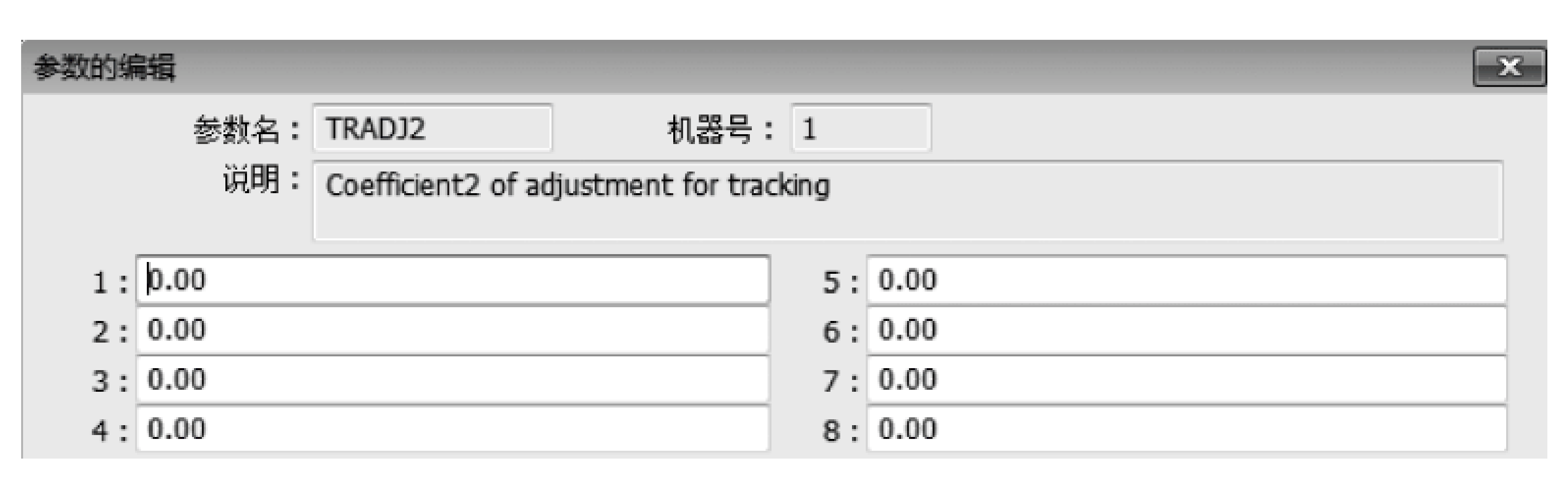


参数名称	参数简写	功　　能	出　厂　值
追踪调节系数 1	TRADJ1	(以传送带速度 100mm/s 为基准)设置一个延迟量。 例 1：当传送带速度＝50mm/s，需要延迟 2mm 时，设置 TRADJ1＝4(2/50×100) 例 2：当传送带速度＝50mm/s，需要延迟 −1mm 时，设置 TRADJ1＝−2(−2/50×100)	

参数的编辑
参数名：TRADJ1　机器号：1
说明：Coefficient1 of adjustment for tracking
1：0.00　5：0.00
2：0.00　6：0.00
3：0.00　7：0.00
4：0.00　8：0.00

续表

参数名称	参数简写	功　　能	出厂值
追踪调节系数2	TRADJ2	修正传送带速度为 Vc＋TRADJ2×（Vc－Vp） Vc＝当前采样的传送带速度 Vp＝前一次采样的传送带速度	



28.10　需要思考的问题

(1) 什么是传送带追踪？什么是视觉追踪？

(2) 传送带追踪使用什么检测器件？

(3) 编码器起什么作用？

(4) TrBase 追踪指令主要有什么用途？

(5) Trk 追踪功能指令的主要用途是什么？

(6) TrWrt 写追踪数据指令的主要用途是什么？

第 29 章　第 29 日——视觉追踪程序的编制

【学习目的】

本章学习追踪程序的编制。追踪程序不是一个单一的程序。追踪程序一般由 5 个程序构成。这 5 个程序不是主程序与子程序的关系，而是各自有独立功能的程序。A 程序是获得机器人与传送带的运动之间的关系。B 程序是获得机器人与视觉系统信息之间的关系。C 程序是获得机器人抓取工作点的数据。CM 程序是实时传递视觉信息的程序。1＃程序是自动运行程序。在本章的学习中要深刻体会各个程序的功用及编制方法。

29.1　追踪程序结构

由于追踪程序不是一个单一的程序，在运行自动程序之前，必须先运行几个采样及标定程序以获得必要的数据。本节叙述不同的追踪程序中所包含的采样程序和自动程序。

如表 29-1 所示的是传送带追踪程序结构所包含的各采样程序和自动程序。

表 29-1　传送带追踪的各采样程序和自动程序

程序名	描　　述	功　　能
A	采样计算每一脉冲机器人移动量	计算每一脉冲机器人移动量
C	工件坐标系与机器人坐标系的配合程序	本程序用于计算工件抓取点坐标值。该坐标值基于光电开关的信号
1	自动程序	本程序用于在追踪移动中吸抓工件并搬运工件
CM1	写数据程序	本程序用于监视编码器值并写入“追踪缓存区”

29.2　视觉追踪程序结构

视觉追踪所包含的各采样标定程序和自动程序如表 29-2 所示。

表 29-2　视觉追踪的各采样程序和自动程序

程序名	描　　述	功　　能
A	计算每一脉冲机器人移动量	计算每一脉冲机器人移动量
B	视觉坐标系与机器人坐标系的标定程序	标定视觉信息坐标与机器人坐标系关系
C	标定“工件抓取点”	本程序用于计算抓取工件的坐标值。该坐标值基于视觉系统的信息
1	自动操作程序	本程序用于自动追踪抓取传送工件
CM1	写数据程序	本程序用于将视觉识别信息及编码器值写入追踪缓存区供 1＃程序使用

各程序之间的关系如图 29-1 和图 29-2 所示。

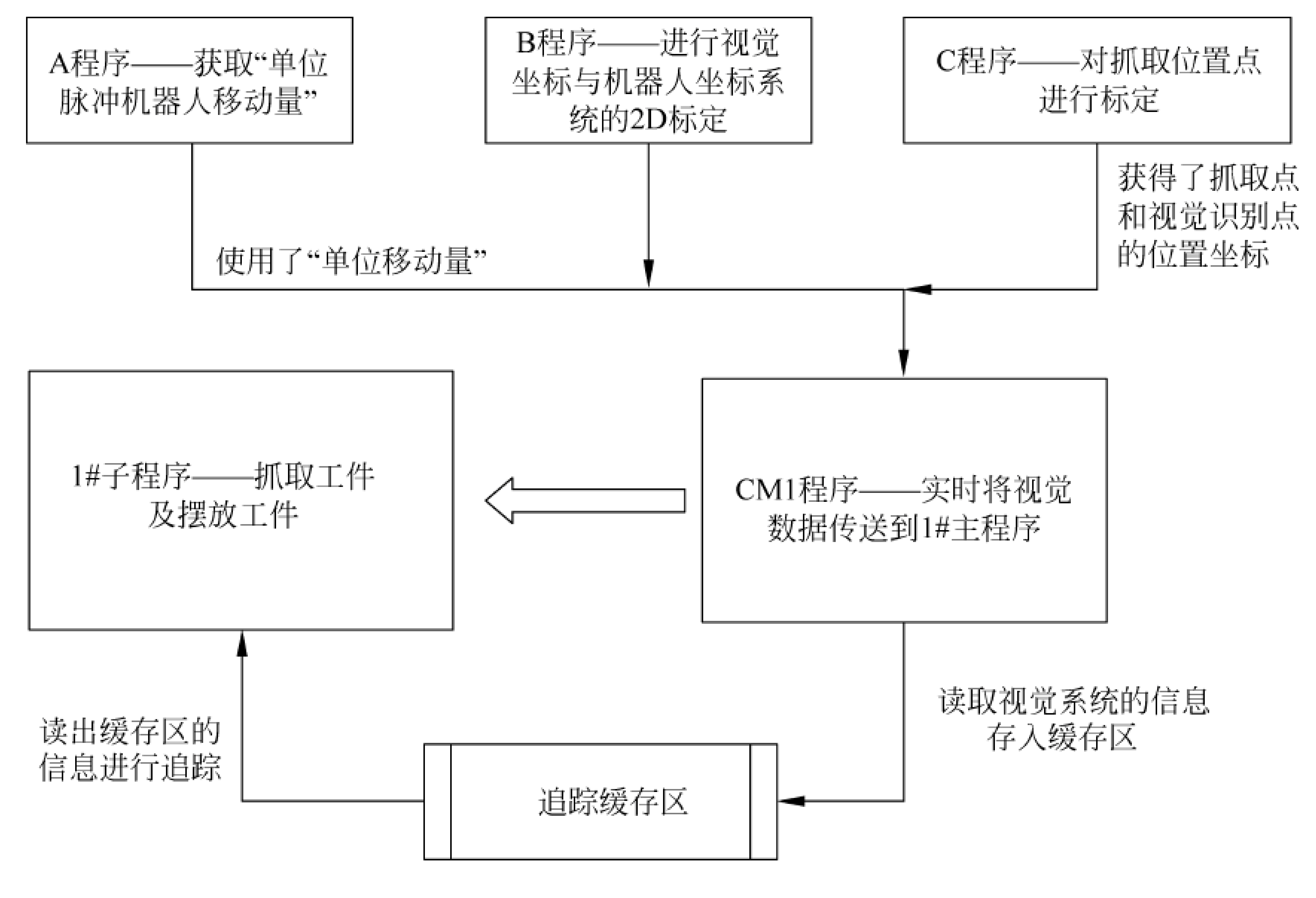


图 29-1　各程序之间的关系图 1

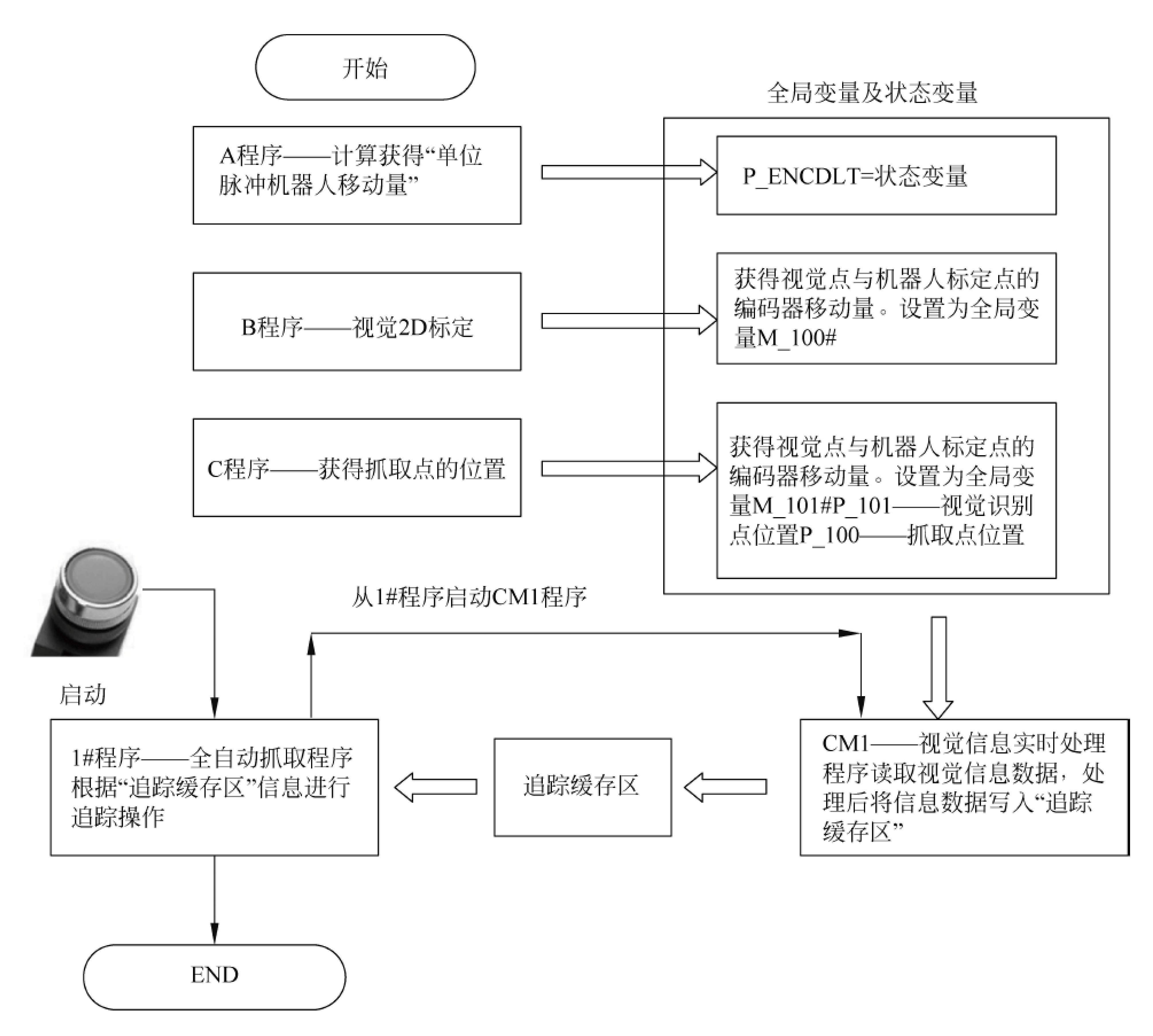


图 29-2　各程序之间的关系图 2

29.3 A 程序——传送带运动量与机器人移动量关系的标定

A 程序用于“传送带移动量”与“机器人移动量”关系的标定。A 程序适用于传送带追踪和视觉追踪。

传送带的标定要参考传送带在机器人坐标系中的移动方向，并且计算“编码器每一脉冲的机器人移动量”。“编码器每一脉冲的机器人移动量”被保存在机器人状态变量 P_EncDlt 中。

在工作前，要监测编码器的数值是否已经输入控制器。转动编码器并监视状态变量 M_Enc(1)～M_Enc(8)，监测这些值是否改变，如果没有改变，要检查参数 TRMODE 的设置是否正确。如果没有设置参数 TRMODE＝1，则 M_Enc(1)的值不会改变。

29.3.1 示教单元运行 A 程序的操作流程

A 程序是采样程序，所以运行时可以用示教单元操作，一步一步地运行程序。也可以使用自动模式运行程序，采集数据并计算。

手动操作程序运行步骤如下。

(1) 在机器人上加装一个标定用的测针（针尖状检具——便于对准工件标记点）。

(2) 设置机器人控制模式为“手动”，设置 TB（示教单元）为“使能状态 ENABLE”（在示教单元背面有使能按钮，按下使能按钮，灯亮后即为使能状态），如图 29-3 所示。

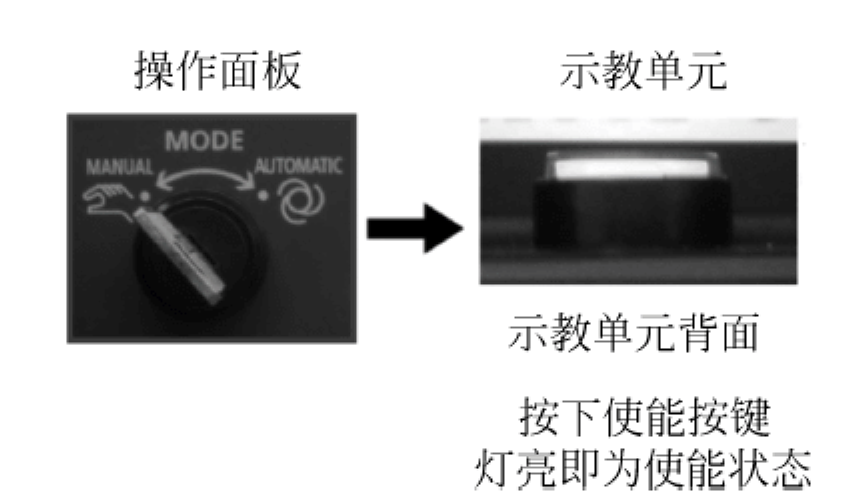


图 29-3 操作模式开关及手动使能开关

(3) 在屏幕上出现< TITLE >时，按下 Exe 键，出现< MENU >屏幕，如图 29-4 所示。

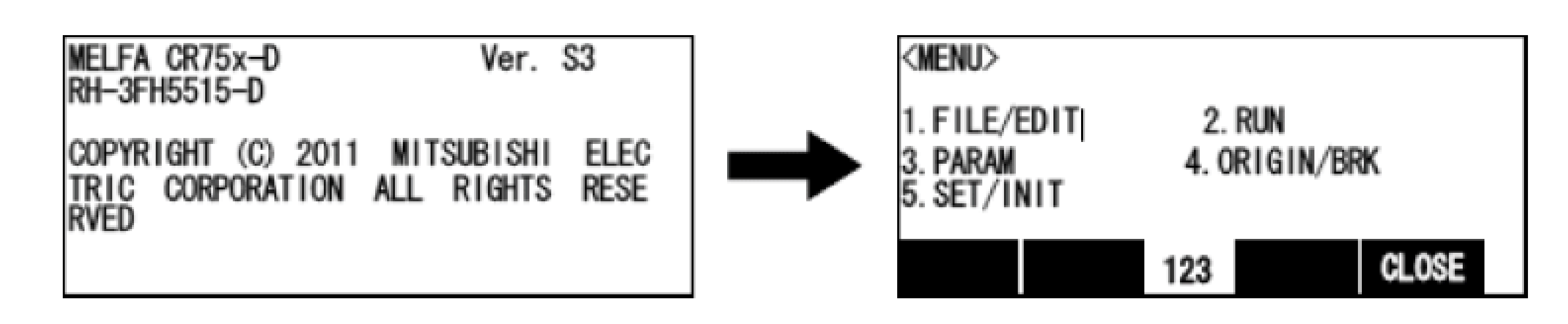


图 29-4 菜单屏幕

(4) 在< MENU >屏上选择“1. FILE/EDIT”，如图 29-5 所示。

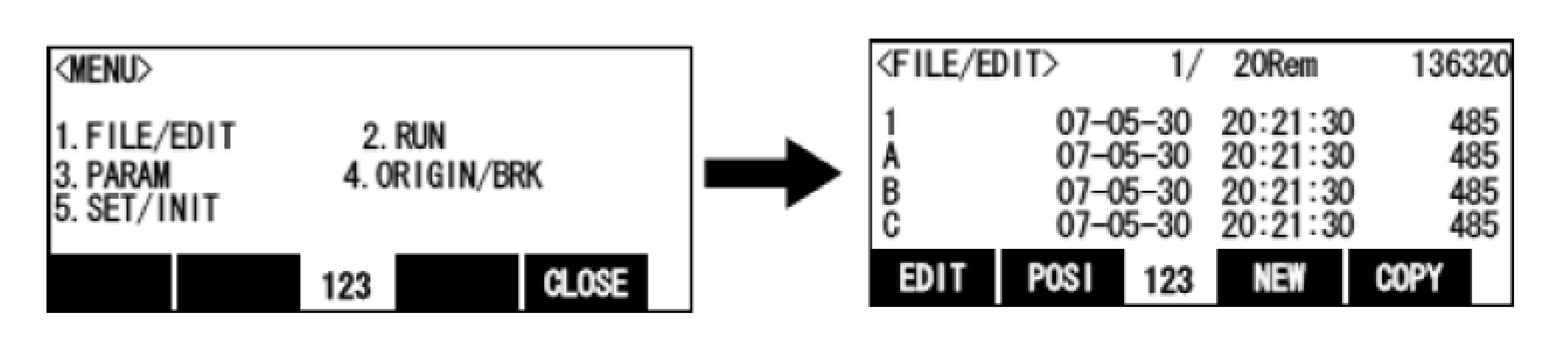


图 29-5 文件及编辑屏幕

（5）选择 A 程序并按 Exe 键，显示< PROGRAM >屏幕，如图 29-6 所示。

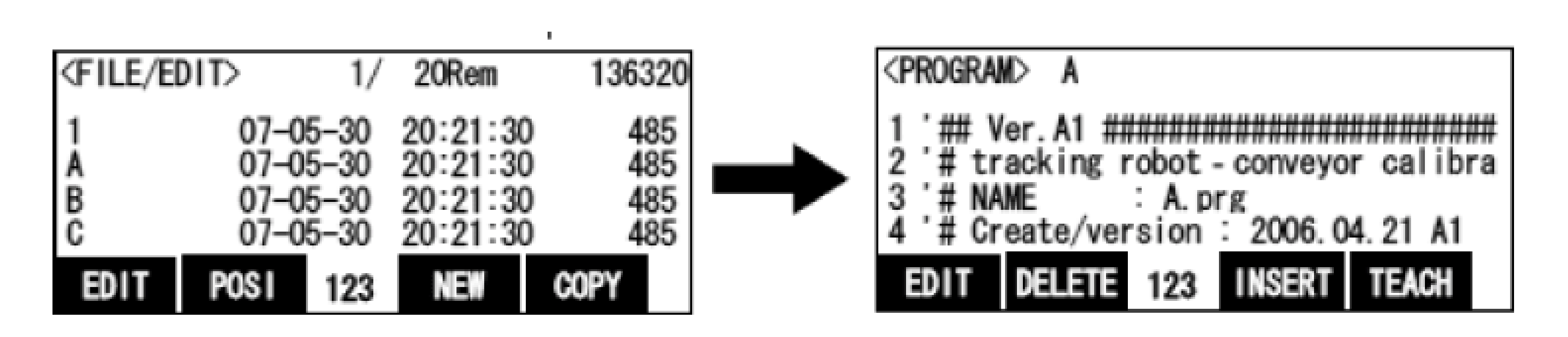


图 29-6　< PROGRAM >及监视屏幕

（6）按下 Function 键，改变 F1～F4 键的功能，如图 29-7 所示。

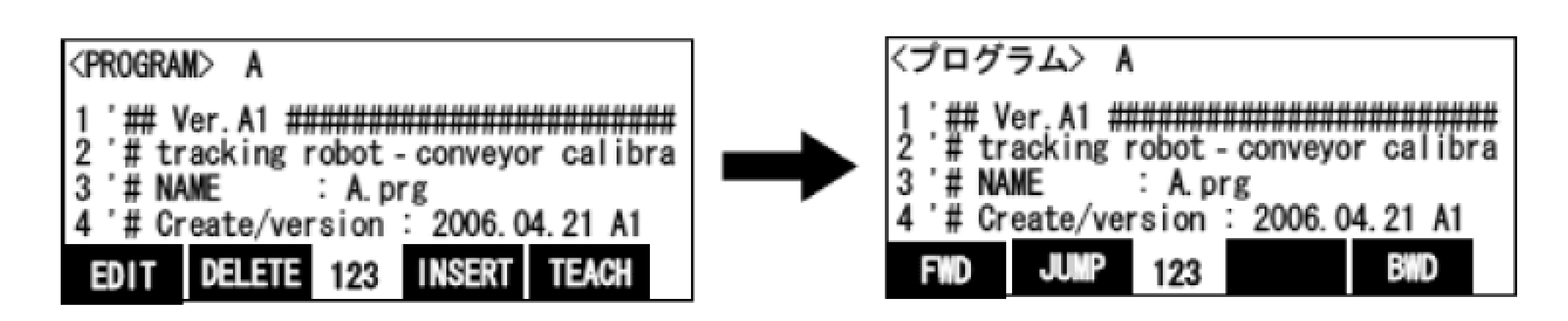


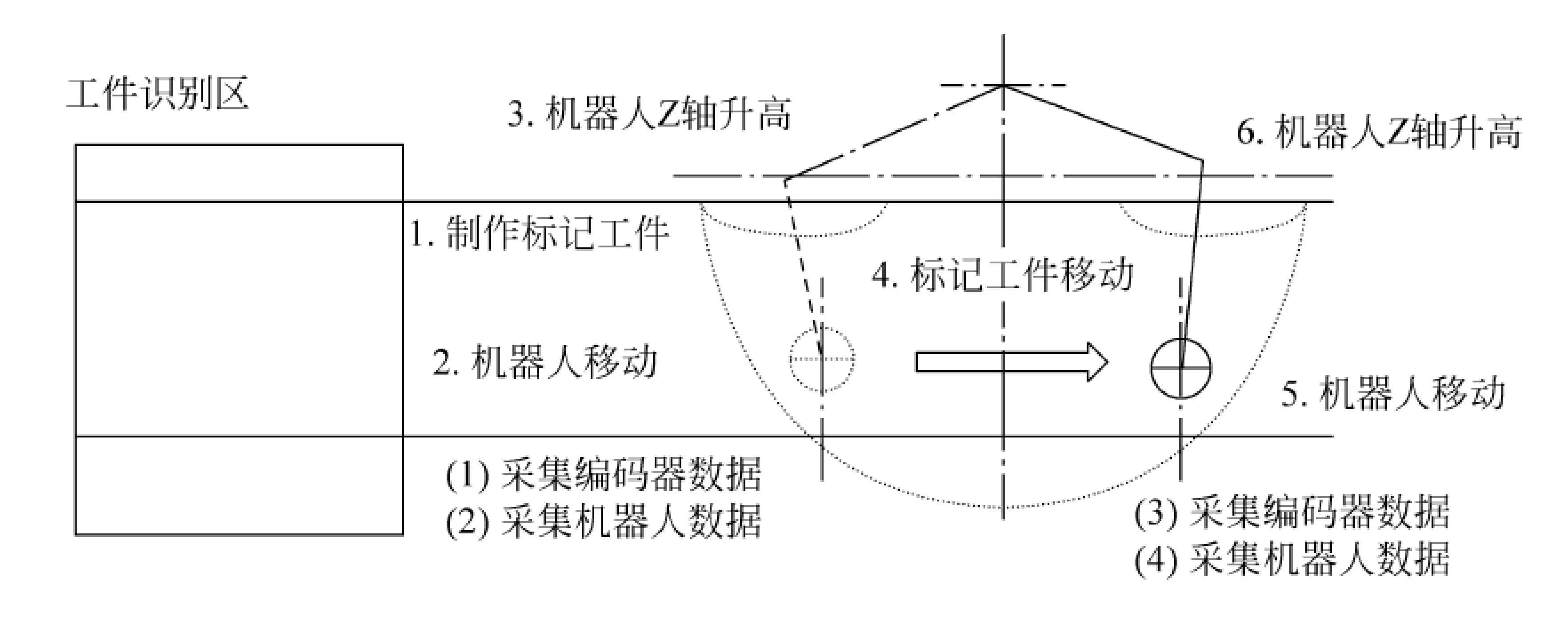
图 29-7　改变功能键

（7）按下 F1(FWD)键，执行步进操作。

以上是使用 TB 单元的步进操作方法，也可以在自动模式下直接执行“启动”或“停止”操作。

29.3.2　设置及操作

A 程序对应的现场操作如图 29-8 所示。



1~6步表示了采样程序的详细步骤

图 29-8　A 程序对应的现场操作

A 程序现场操作步骤如下。

在机器人上装一个标定用的测针——针尖状检具便于对准工件标记点，如图 29-9 所示。

（1）在 RT 软件中，设置“任务区 1”内的程序为 A 程序。

（2）给工件贴上识别标记；移动工件进入追踪区，并停止传送带。工件停止在“第 1 点”。可以使用的标记板如图 29-10 所示（可使用其中一个标记点作为“工件标记点”）。

（3）测量第 1 点数据——选择“手动”模式，使用 TB 单元，移动机器人对准追踪区内第 1 点。

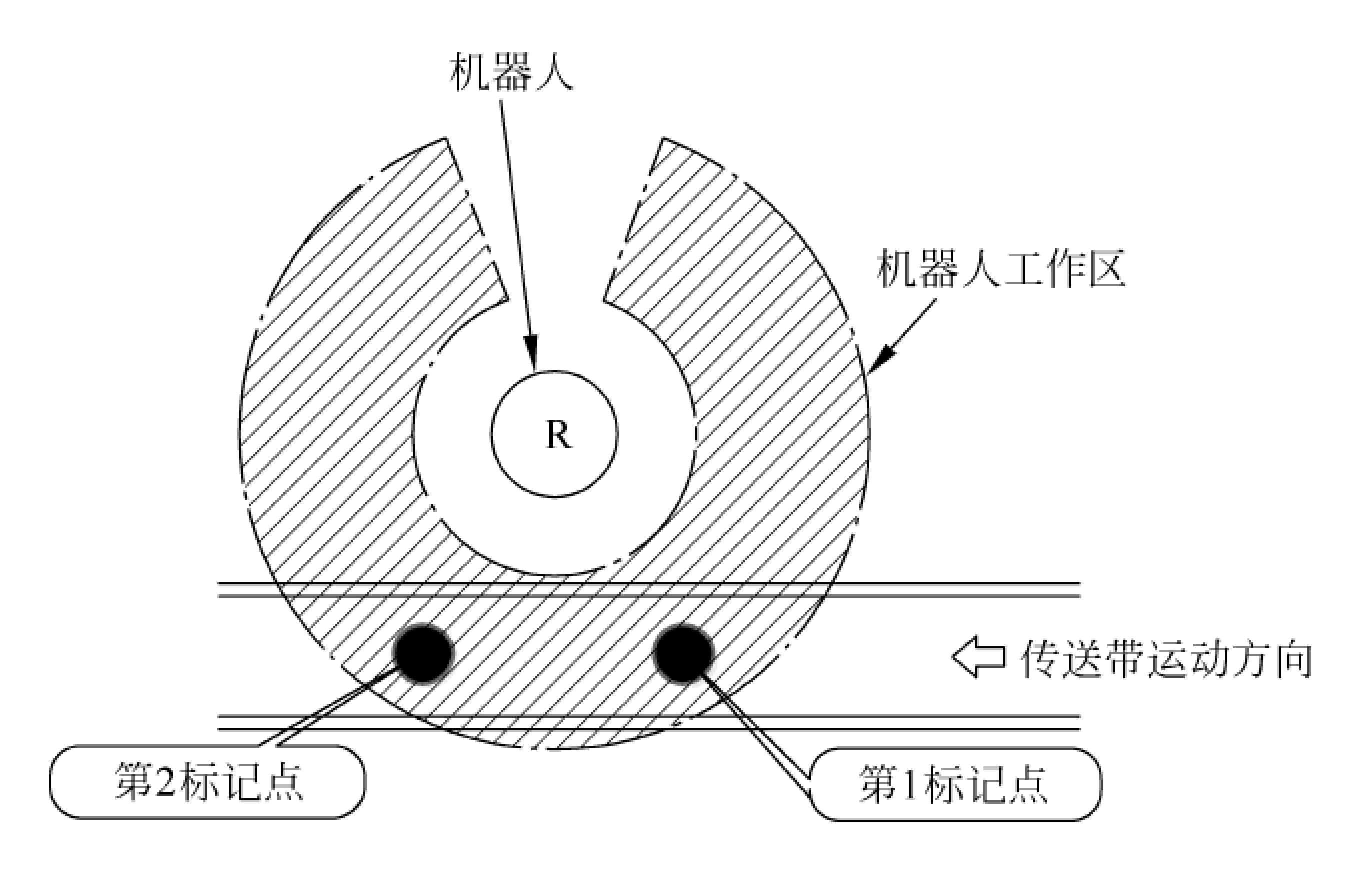


图 29-9　第 1 标记点和第 2 标记点

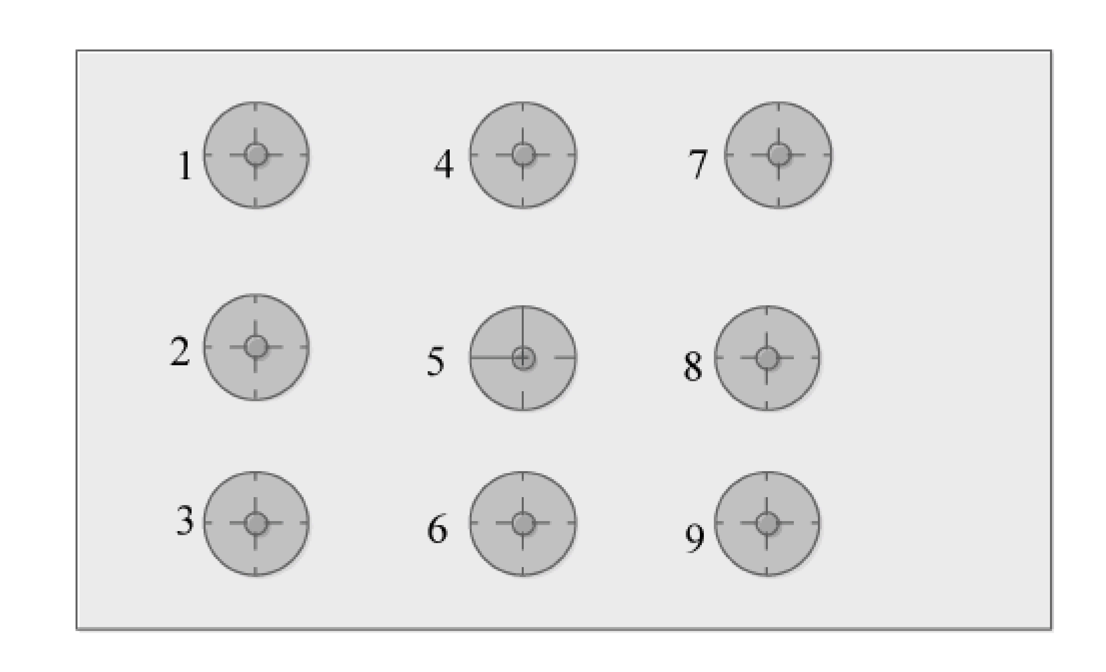


图 29-10　可以在 A 程序中使用的“标记板”

(4) 选择“自动”模式，使用 TB 示教单元启动 A 程序，A 程序自动运行然后停止在 HLT 步。

(5) 选择“手动”模式，使机器人 Z 方向上升。

(6) 启动传送带使其带动工件移动到第 2 点，停止。

(7) 选择“手动”模式，使用 TB 单元，移动机器人对准追踪区内第 2 点(注意：标记工件移动范围应该尽可能达到最大值，这样可以减少测量误差)。在“手动”期间，不要使程序复位。

(8) 选择“自动”模式，再次启动 A 程序，A 程序自动运行然后停止在 END 步。

(9) 使机器人 Z 方向上升。

(10) 完成 A 程序标定。

在执行 A 程序过程中，自动读取了第 1 点、第 2 点的编码器数据、位置数据，并进行标定。

29.3.3　确认 A 程序执行结果

在 RT 软件中监视 P_EncDlt 变量值。这个变量显示每一脉冲机器人各坐标移动量。例如，如果该值仅显示 Y 轴为 0.5，则表示传送带移动量为 100 脉冲时，机器人在 Y 轴方向移动 50mm(0.5×100=50)。

29.3.4 多传送带场合

多传送带场合按上述方式同样操作，但要注意以下要点。

例如，使用2传送带，编码器编号=2。

(1) 在对应传送带2的A程序中，将PE变量的X坐标设置为2。

(2) 在动作确认操作中，使用RT ToolBox2软件，检查变量P_EncDlt(2)的数值。

29.3.5 A程序流程图

A程序流程图如图29-11所示。

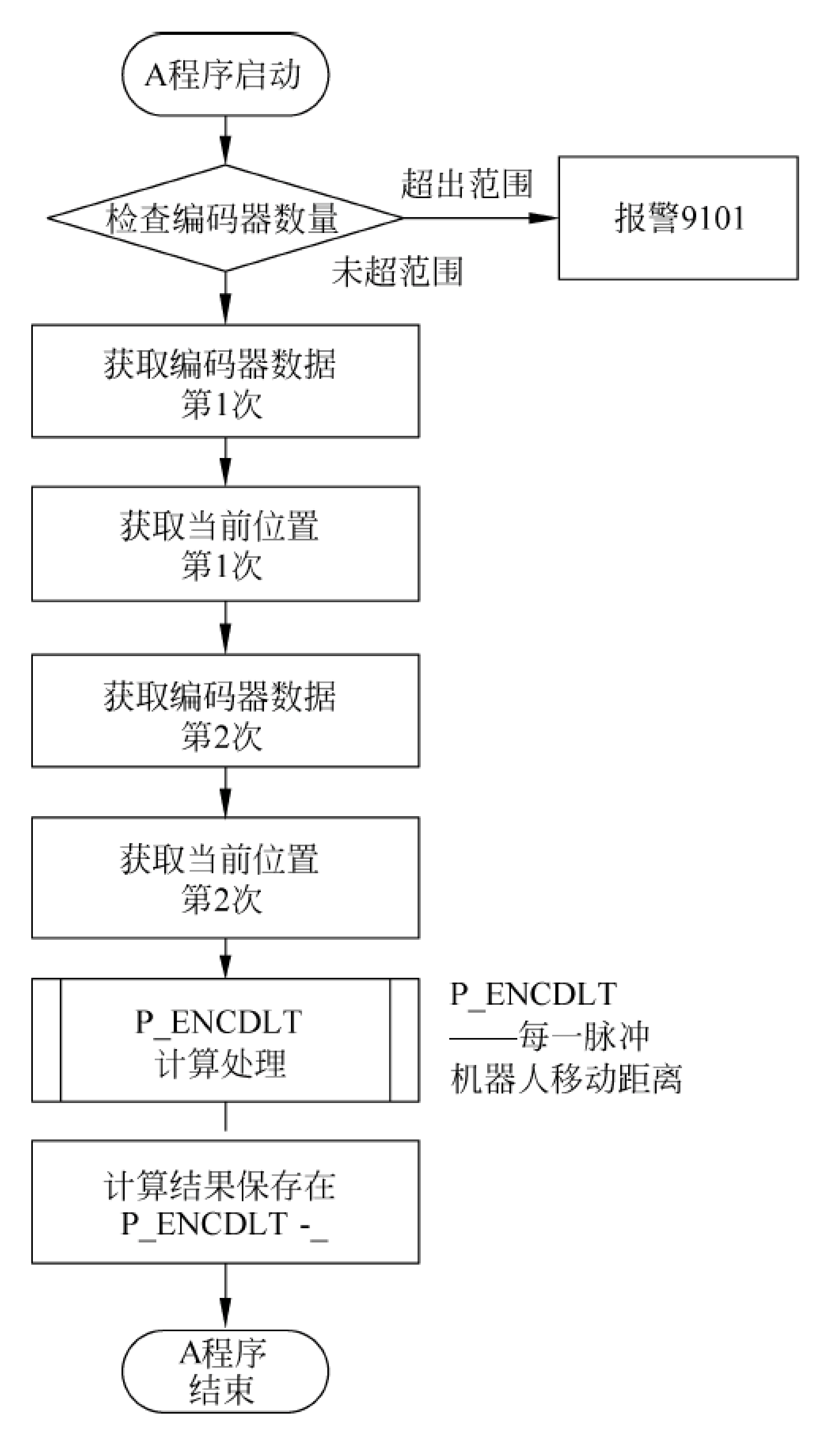


图29-11 A程序流程图

29.3.6 实用A程序

```
'——(将编码器编号设置为"PE"变量的X坐标)
'——检查设置变量。
MECMAX = 8 '——设置编码器编号最大值MECMAX = 8。
If PE.X < 1 Or PE.X > MECMAX Then Error 9101'——如果编码器编号超范围，就报警9101。
MENCNO = PE.X'——(获取编码器编号)将PE.X值赋予MENCNO。
```

```
6 '——在传送带上放置带标记的工件,移动工件进入追踪区。
7 '——移动机器人到标记工件的中心点。
8 MX10EC1# = M_Enc(MENCNO) '——获取此位置编码器数据(第 1 点)。
9 PX10PS1 = P_Zero'——清零。
10 PX10PS1 = P_Fbc(1) '——获取机器人当前值。PX10PS1 = 机器人当前值(第 1 点)。
11 HLT '—— 暂停。
12 '——使机器人向上运动。
13 '——移动机器人到第 2 点标记工件中心位置。
14 MX10EC2# = M_Enc(MENCNO) '——获取编码器值,第 2 次。
15 PX10PS2 = P_Zero '——清零。
16  PX10PS2 = P_Fbc(1) '——获取当前值,第 2 次; PX10PS2 = 当前值(第 2 次)。
17 '——使机器人向上运动。
19 GoSub * S10ENC '——跳转到进行 P_EncDlt 计算的子程序。
20 P_EncDlt(MENCNO) = PY10ENC '——保存计算结果。
21 End'——主程序结束。
23 '##### 计算 P_EncDlt 的程序#####
24 '——MX10EC1:第 1 次编码器数据
25 '——MX10EC2:第 2 次编码器数据
26 '——PX10PS1: 第 1 次位置数据
27 '——PX10PS2:第 1 次位置数据
28 '——PY10ENC:计算结果
29 * S10ENC'——    P_EncDlt 计算子程序。
30 M10ED# = MX10EC2# - MX10EC1# '——(编码器数据相减)
31 If M10ED#>800000000.0 Then M10ED# = M10ED# - 1000000000.0
32 If M10ED#<-800000000.0 Then M10ED# = M10ED# + 1000000000.0
'——以上是对编码器数据的处理。
33 PY10ENC.X = (PX10PS2.X - PX10PS1.X)/M10ED#
34 PY10ENC.Y = (PX10PS2.Y - PX10PS1.Y)/M10ED#
35 PY10ENC.Z = (PX10PS2.Z - PX10PS1.Z)/M10ED#
36 PY10ENC.A = (PX10PS2.A - PX10PS1.A)/M10ED#
37 PY10ENC.B = (PX10PS2.B - PX10PS1.B)/M10ED#
38 PY10ENC.C = (PX10PS2.C - PX10PS1.C)/M10ED#
39 PY10ENC.L1 = (PX10PS2.L1 - PX10PS1.L1)/M10ED#
40 PY10ENC.L2 = (PX10PS2.L2 - PX10PS1.L2)/M10ED#
'——以上是计算"一个编码器脉冲对应的机器人移动量"的过程。各坐标值相减的结果除以编码器数据。
41 Return'——子程序结束。
43 '以下是 A 程序运行前必须预设置的变量。
PE = (1.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
PE.X 是编码器编号,是预先设置的位置变量形式。
PX10PS1 = (0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)(第 1 位置数据)
PX10PS2 = (0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0) (第 2 位置数据)
PY10ENC = (0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)("一个编码器脉冲对应的机器人移动量")
```

从以上 A 程序可知,在机器人工作区域内取两个工作点(一般应该涵盖机器人工作区最大区域),分别获取这两点的编码器数值和位置点数据。通过计算获得“机器人移动量/每脉冲”。

29.4 B程序——视觉坐标与机器人坐标关系的标定

本节介绍使用B程序进行的标定。B程序用于视觉系统与机器人坐标系的标定。

29.4.1 示教单元的操作

(1) 使用T/B打开B程序。

(2) 设置控制模式为“手动”,设置TB为使能状态ENABLE,如图29-12所示。

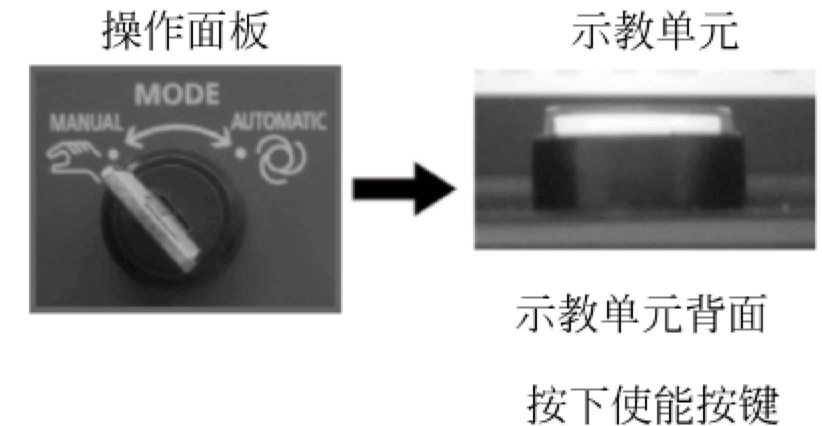


图29-12 操作模式开关和手动使能开关

(3) 在屏幕上出现< TITLE >时,按下Exe键,出现< MENU >屏幕,如图29-13所示。

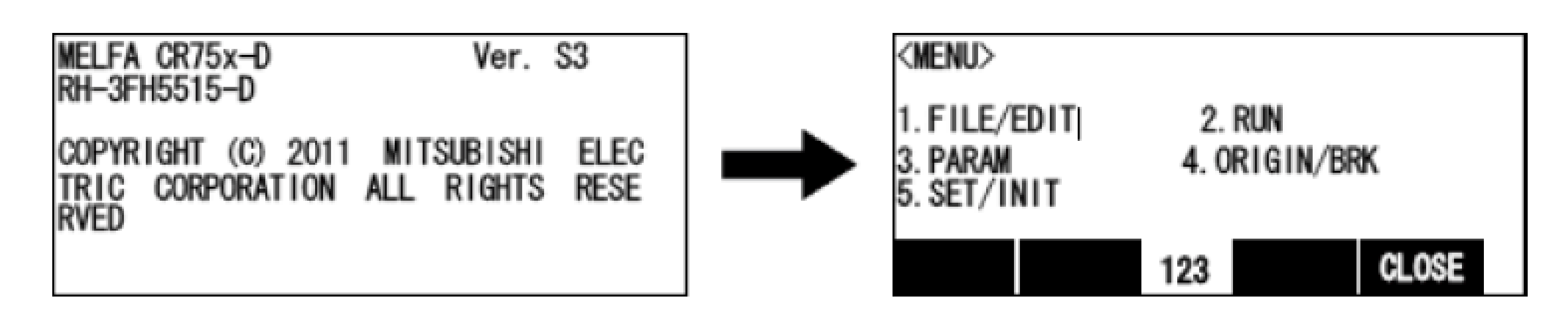


图29-13 菜单屏幕

(4) 选择“1. FILE/EDIT”,如图29-14所示。

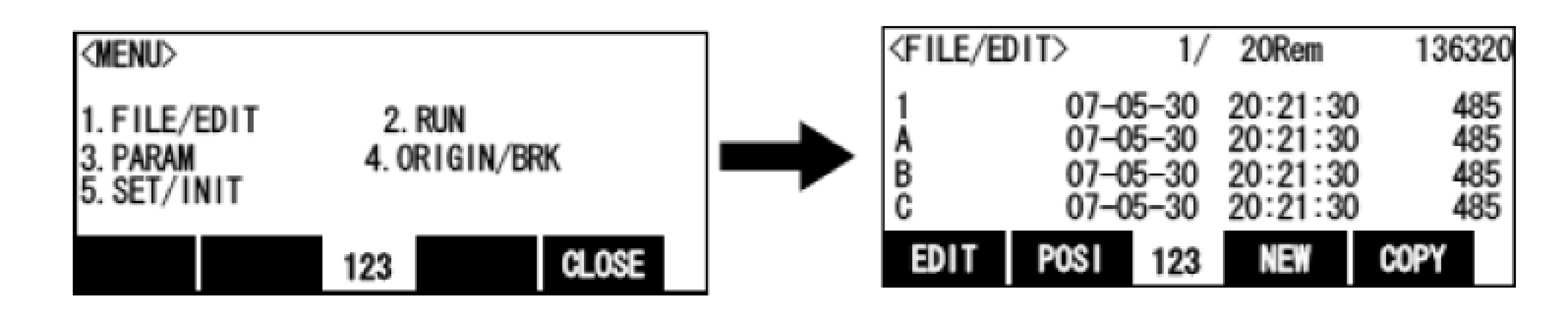


图29-14 文件及编辑屏幕

(5) 选择B程序并按Exe键,显示< PROGRAM >屏幕,如图29-15所示。



图29-15 < PROGRAM >屏幕

(6) 按下Function键,改变功能显示,如图29-16所示。

(7) 按下F1(FWD)键,执行步进操作。

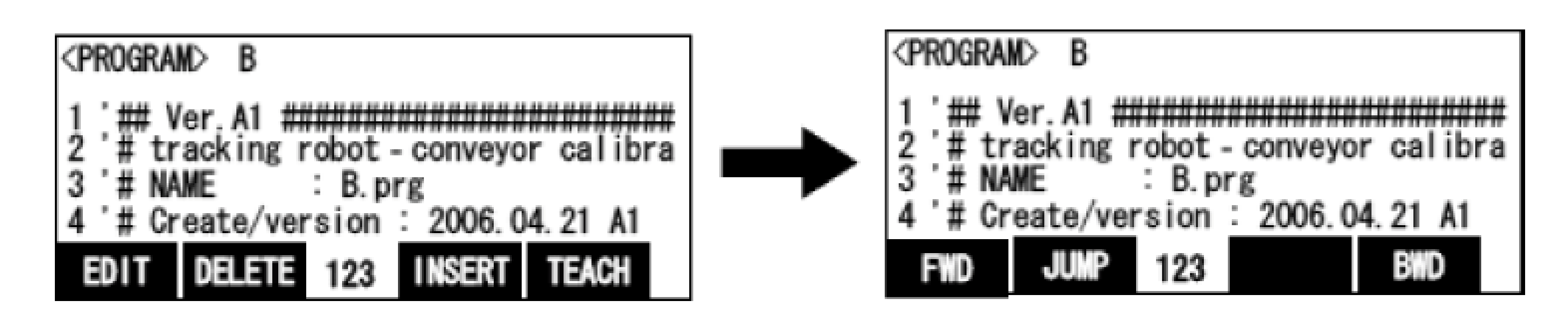


图 29-16　改变操作键功能

29.4.2　现场操作流程

根据图 29-17～图 29-19 所示，进行的操作如下。

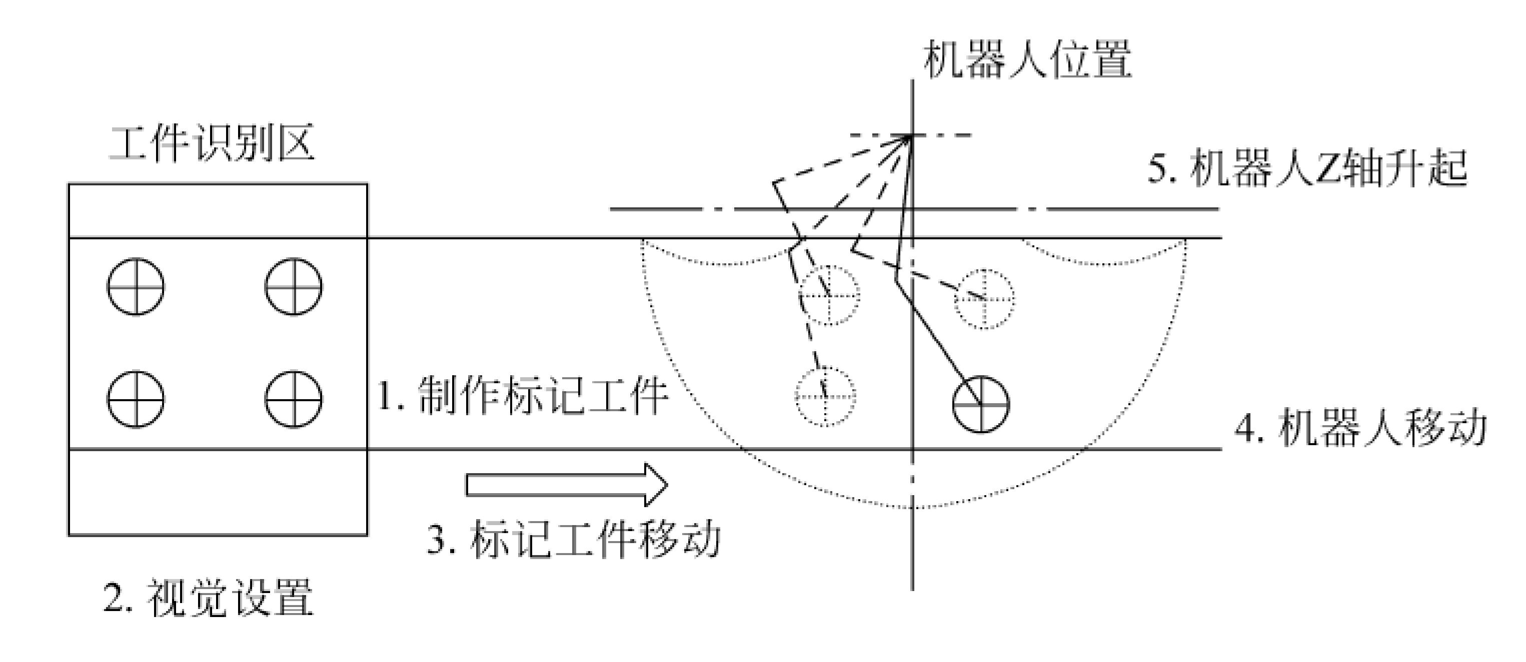


图 29-17　B 程序操作图示

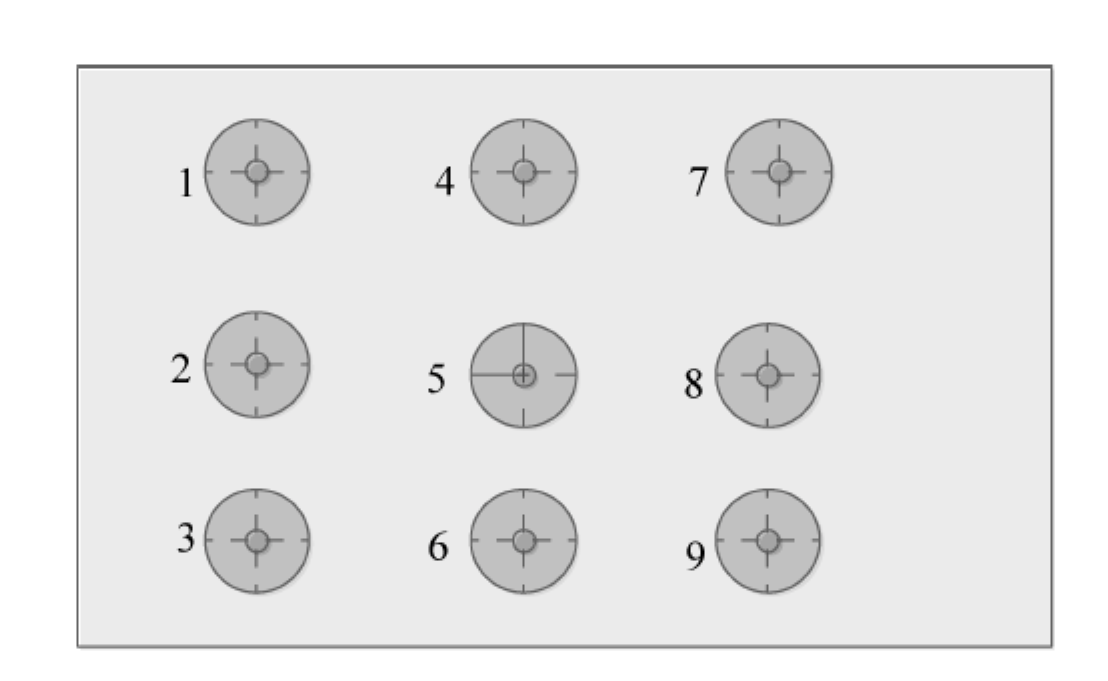


图 29-18　B 程序使用的标定板

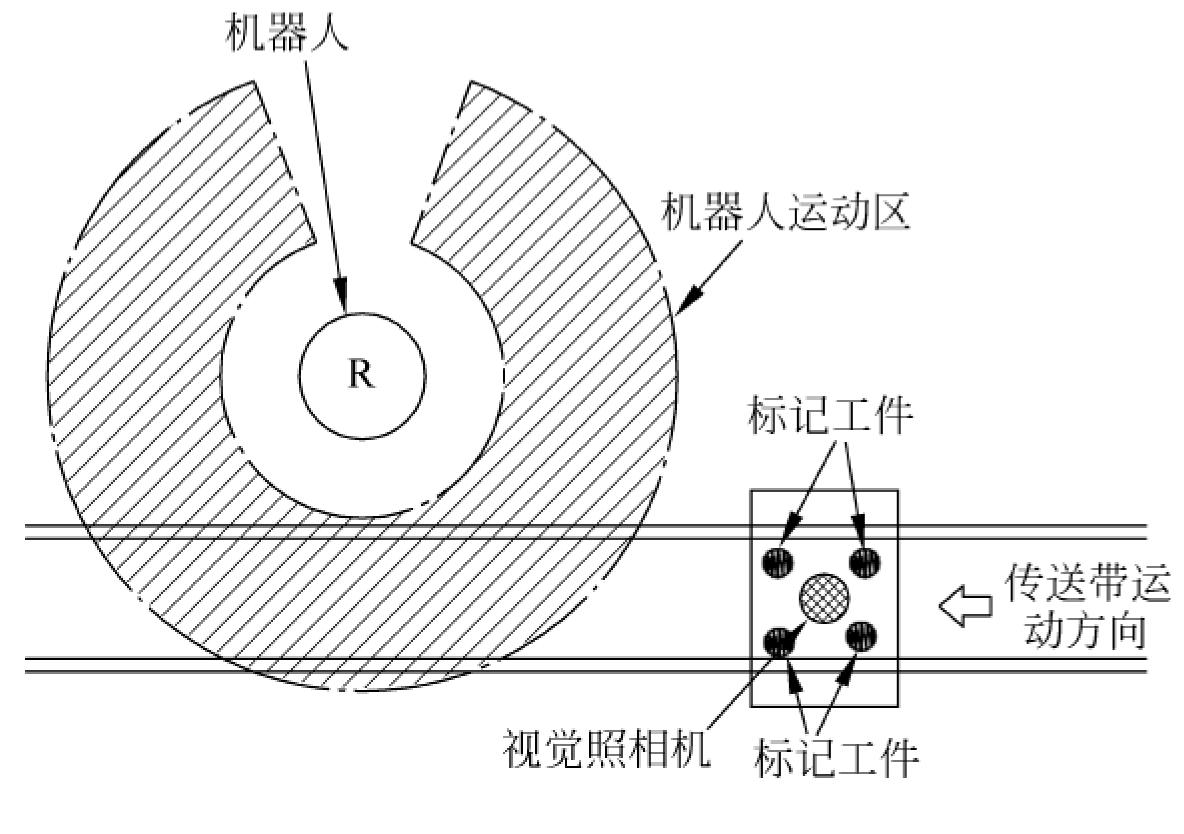


图 29-19　B 程序工作示意图

(1) 在机器人上加装测针。

(2) 制作标记板如图 29-18 所示。

(3) 在 RT 软件中,设置“任务区 1”中为“B 程序”。

(4) 检查与视觉系统的通信状态,保证通信正常。启动视觉系统。

(5) 启动传送带。

(6) 选择机器人“自动工作”模式。

(7) 启动 B 程序(可使用 TB 单元)。

(8) 放置“标记板”在传送带上使“标记板”通过视觉区一直运行到“机器人工作区”,在“机器人工作区”中部使传送带停止。

(9) 由视觉系统获得标记板 4 个以上点位数据(如 1、3、5、7、9 点位)。

(10) B 程序停止在 HLT 步。

(11) 选择“手动模式”。

(12) 通过示教获得 4 个以上的点位数据(例如 1、3、5、7、9 点位),全部记录在 RT 软件的 2D 标定界面,如图 29-20 所示。

(13) 机器人 Z 轴上升。

(14) 选择“自动模式”,启动 B 程序直到执行到 END 步。

(15) 在 RT 软件的 2D-标定界面写入视觉系统检测获得的数据,进行标定并写入机器人。

B 程序执行完毕。

29.4.3 操作确认

(1) 使用 RT 软件检查变量 M_100()。在 RT 软件的“程序监视”画面,通过“变量监视”功能,也可以监视 M_100()的数值。M_100()为编码器数据差。

(2) 确认在视觉传感器侧获得的编码器数据与机器人侧获得的编码器数据之差已经设置在 M_100()。

29.4.4 实用 B 程序

```
'——将"编码器编号"设置为变量 PE 的 X 坐标。
'——检查设定值。
 MECMAX = 8 '——设置编码器编号的最大值。
 If PE.X < 1 Or PE.X > MECMAX Then Error 9101 '——如果编码器编号超出允许范围,则报警,否则执行下一步。
 MENCNO = PE.X '——取得编码器编号。
'——在视觉传感器识别区域放置标定板。
'——观察视觉画面,确认标定板位置是否正确。
'——视觉系统照相,由视觉系统记录 5 个点的数据(视觉数据)。
 ME1# = M_Enc(MENCNO) '——取得编码器数据(第 1 次)。
Hlt'——暂停。
'——将标定板由传送带移动至机器人动作范围内。
'——将机器人抓手移动至标记点 1 的中心处,获取机器人坐标。
'——(8)重复作业,获取 5 个点机器人位置。
'——(10)上升机器人手臂。
```

```
ME2# = M_Enc(MENCNO) '——取得编码器数据(第 2 次)。
MED# = ME1# - ME2#   '——计算两个位置的编码器数值差。
If MED#>800000000.0# Then MED# = MED# - 1000000000.0#
If MED#<-800000000.0# Then MED# = MED# + 1000000000.0#
M_100#(MENCNO) = MED#
'——以上是对编码器数据的处理。
Hlt'—— 暂停。
End'——主程序结束。
'——以下是运行前必须预置的变量。
PE = (+1.000, +0.000, +0.000, +0.000, +0.000, +0.000, +0.000, +0.000)(0,0)
```

B 程序执行完毕后获得两点之间的“编码器移动量 MED#”，设置为全局变量 M_100#。

29.4.5 2D 标定操作

操作步骤如下。

(1) 打开 RT 软件，单击“维护”→2D，弹出如图 29-20 所示界面。

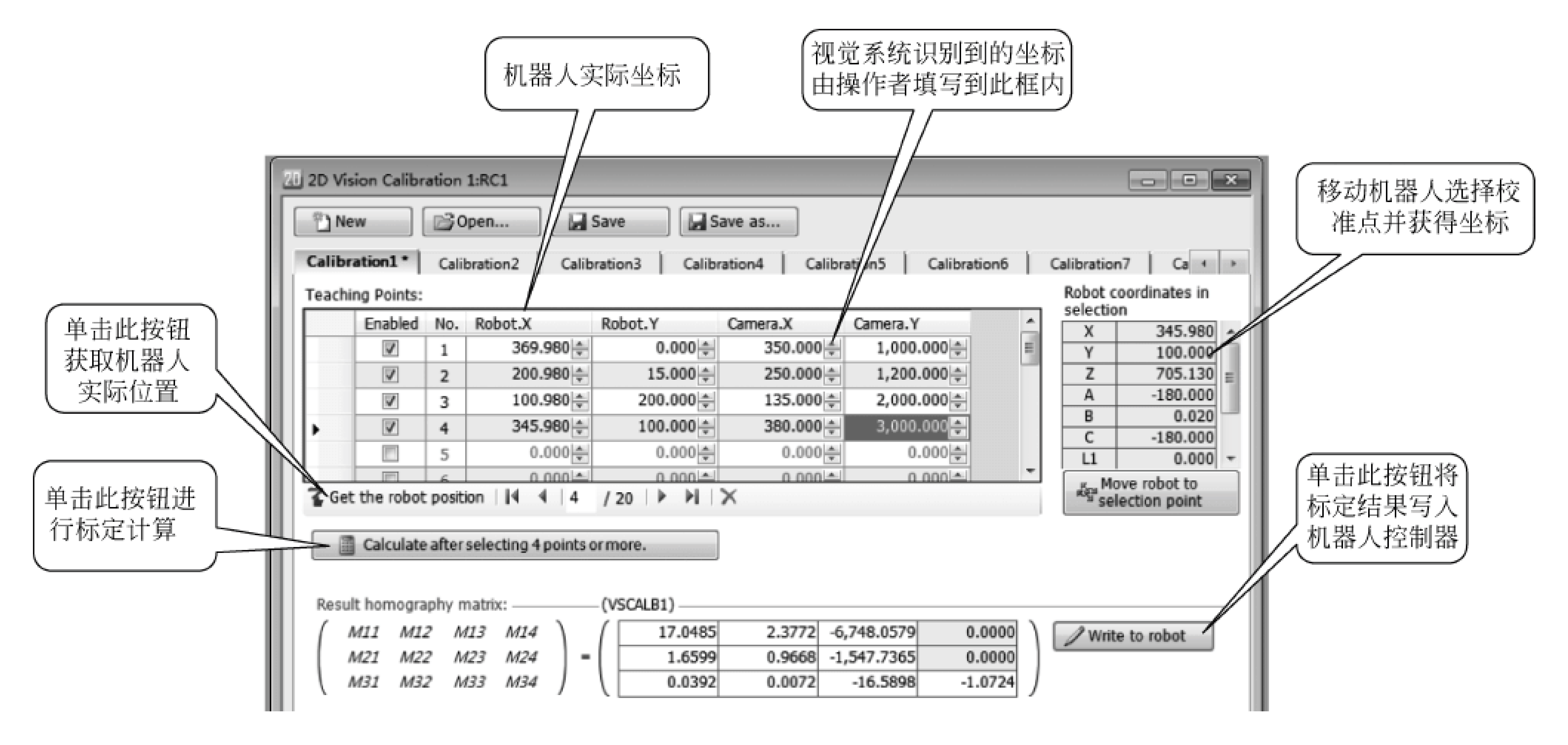


图 29-20 RT 软件的 2D 标定界面

(2) 每示教一点就按“获取机器人数据”键，该示教点数据自动进入“机器人坐标”框。一共获取 5 点数据。

(3) 将视觉系统获得的对应 5 点数据写入“视觉坐标”框。

(4) 按“数据计算标定”键，数据进行标定转换。

(5) 按“写入机器人”键。

(6) 操作完成。

29.5 C 程序——抓取点标定

本节介绍由 C 程序执行的任务。在传送带追踪和视觉追踪都需要执行 C 程序，只是各自的执行方法有所不同。

29.5.1 用于传送带追踪的程序

在用于传送带追踪的C程序中，既获取了光电开关检测点的编码器数据，也获取了机器人抓取工件点的数据，因此在光电开关检测动作时，机器人能够识别工件坐标。

1. 示教单元运行C程序的操作流程

(1) 设置控制模式为“手动”，设置TB为使能状态ENABLE，操作如图29-21所示。

(2) 在手动模式下，进行示教操作，运动到“抓取点”“待避点”“下料摆放点”。

(3) 当选择“自动模式”时，可使用示教单元执行“启动”，停止操作。

操作面板
示教单元
示教单元背面
按下使能按键
灯亮即为使能状态

图 29-21 模式开关及使能开关操作

2. C程序操作流程

图29-22是C程序操作图示。

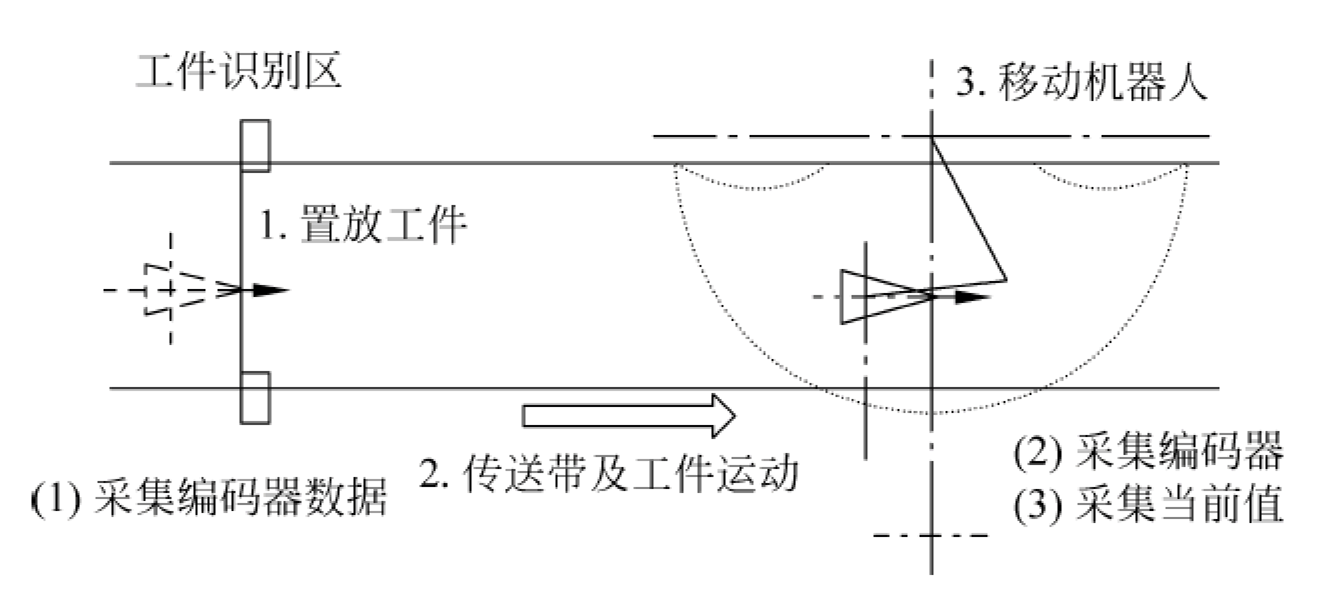


图 29-22 C程序操作图示

C程序操作流程如下。

(1) 在机器人上加装测针(与A程序相同)。

(2) 选择一工件，在工件上做出标记。

(3) 在RT软件中，设置“任务区1”中为“C程序”。

(4) 启动传送带；移动工件到“光电开关”处，停止传送带。

(5) 选择机器人“自动工作”模式。

(6) 启动C程序(可使用TB单元)，一直运行C程序到HLT暂停步。

(7) 启动传送带，在“机器人工作区”中部使传送带停止。

(8) 选择“手动模式"。

(9) 使用机器人示教功能，标定机器人抓取点PGT。

(10) 选择“自动模式”，启动C程序直到执行到END步。

(11) 标定出“待避点P1”“下料摆放点PPT”。

(12) C程序执行完毕。

3. 动作确认

使用RT软件监视确认变量M_101()，P_100()和P_102()的数值。

(1) M_101：在“光电开关动作点”与“机器人抓取点”编码器数据之间的差值。

(2) P_100()：工件被抓取点的位置。

（3）P_102()：在 STEP1 中设置的 PRM1 数值。

必须检查以上各值是否正确。

4. 实用 C 程序

1）流程图

C 程序流程图如图 29-23 所示。

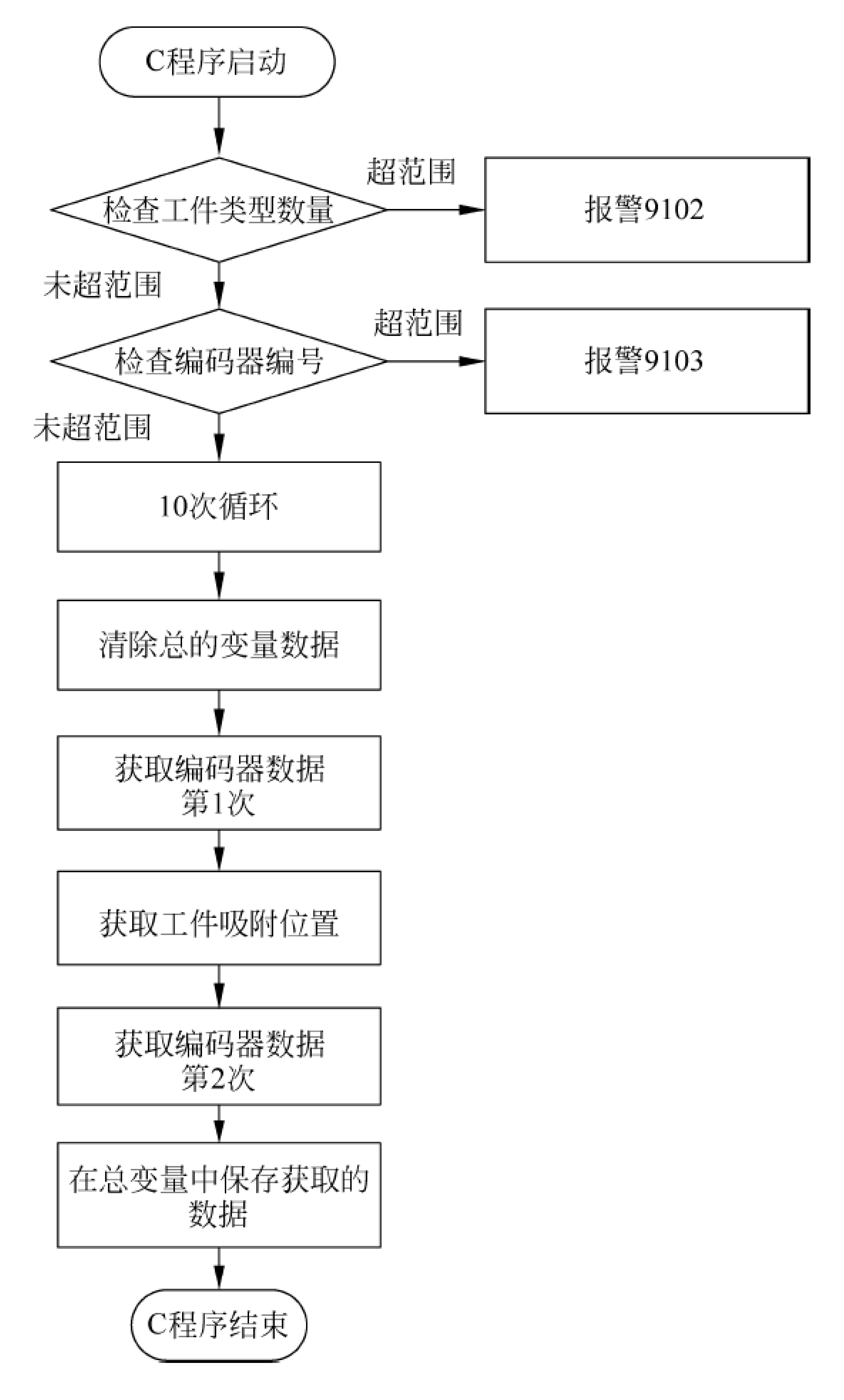


图 29-23　C 程序流程图

2）C 程序

```
'——以变量 PRM1 的 X 坐标值表示工件类型数量。
'—— 以变量 PRM1 的 Y 坐标值表示编码器编号。
'——以变量 PRM1 的 Z 坐标值表示传感器数量。
'——检查设置在 PRM1 中的数据。
 MWKMAX = 10 '—— 设置工件类型数最大 = 10。
 MECMAX = 8 '—— 设置编码器编号最大 = 8。
 MWKNO = PRM1.X '—— 获取工件类型数量。
 MENCNO = PRM1.Y'——获取编码器编号数量。
If MWKNO < 1 Or MWKNO > MWKMAX Then Error 9102
'——如果工件类型数量超范围就报警 9102
```

```
10 If MENCNO < 1 Or MENCNO > MECMAX Then Error 9101'——如果编码器编号数超范围就报警 9101。
11 For M1 = 1 TO 10 '——做一循环,对相关变量清零。
12 P_100(M1) = P_Zero'——对存储工件位置的变量置零。
13 P_102(M1) = P_Zero'——对存储操作条件的变量置零。
14  M_101#(M1) = 0 '——对存储编码器数值差的变量置零。
15 Next M1'——下一循环。
16 '——将工件移动到光电开关位置。
17 ME1# = M_Enc(MENCNO) '——获取编码器数据(第 1 次)。
18  HLT'——暂停。
19 '—— 移动传送带上的工件到机器人工作区。
     移动机器人到抓取点位置。
20 ME2# = M_Enc(MENCNO) '——获取编码器数值(第 2 次)。
21 P_100(MWKNO) = P_Fbc(1) '——获取工件抓取点数据(当前位置)。
22 '——执行步进操作直到结束。
23 MED# = ME2# - ME1#  '——计算两次编码器的差值。MED# = 两次编码器数据的差值。
24 If MED# > 800000000.0 Then MED#  =  MED# - 1000000000.0
25 If MED# < - 800000000.0 Then MED#  =  MED# + 1000000000.0
26 '——以上是对编码器数值的处理。
27  M_101#(MWKNO) = MED# '—— 将编码器的数值差 MED#保存在一个全局变量 M_101#中。
28  P_102(MWKNO).X = PRM1.Y '——保存编码器编号数在一个全局变量中。
29 P_102(MWKNO).Y = PRM1.Z '——保存传感器数量在一个全局变量中。
30 End'——程序结束。
```

29.5.2 用于视觉追踪的 C 程序

视觉追踪 C 程序获取由视觉传感器识别的工件位置点的编码器数据以及机器人抓取工件点的数据,这样机器人就能够识别由视觉系统获取的工件坐标。下面解释工作流程和相关工作内容术语。

1. 示教单元手动操作

(1) 设置控制模式为"手动",设置 TB 为使能状态 ENABLE,如图 29-24 所示。

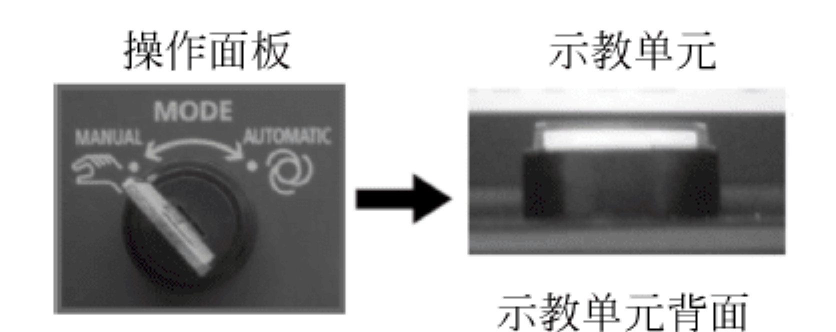


图 29-24　模式开关及使能开关操作

(2) 在手动模式下,进行示教操作,运动到"抓取点""待避点""下料摆放点"。

(3) 当选择"自动模式"时,可使用示教单元执行"启动",停止操作。

2. 执行 C 程序操作的流程

(1) 操作流程,如图 29-25 和图 29-26 所示。

① 在机器人上加装测针(与 A 程序相同)。

② 制作标记工件,选择一个工件,在工件上做出标记。

③ 在 RT 软件中,设置"任务区 1"中为"C 程序"。

④ 检查与视觉系统的通信状态,保证通信正常。启动视觉系统。

⑤ 启动传送带(如果是"面成像照相视觉系统",可以在静态下拍照,则不需要启动传送带。如果是"线扫描成像系统",则需要启动传送带)。

⑥ 选择机器人"自动工作"模式。

⑦ 启动 C 程序(可使用 TB 单元)。

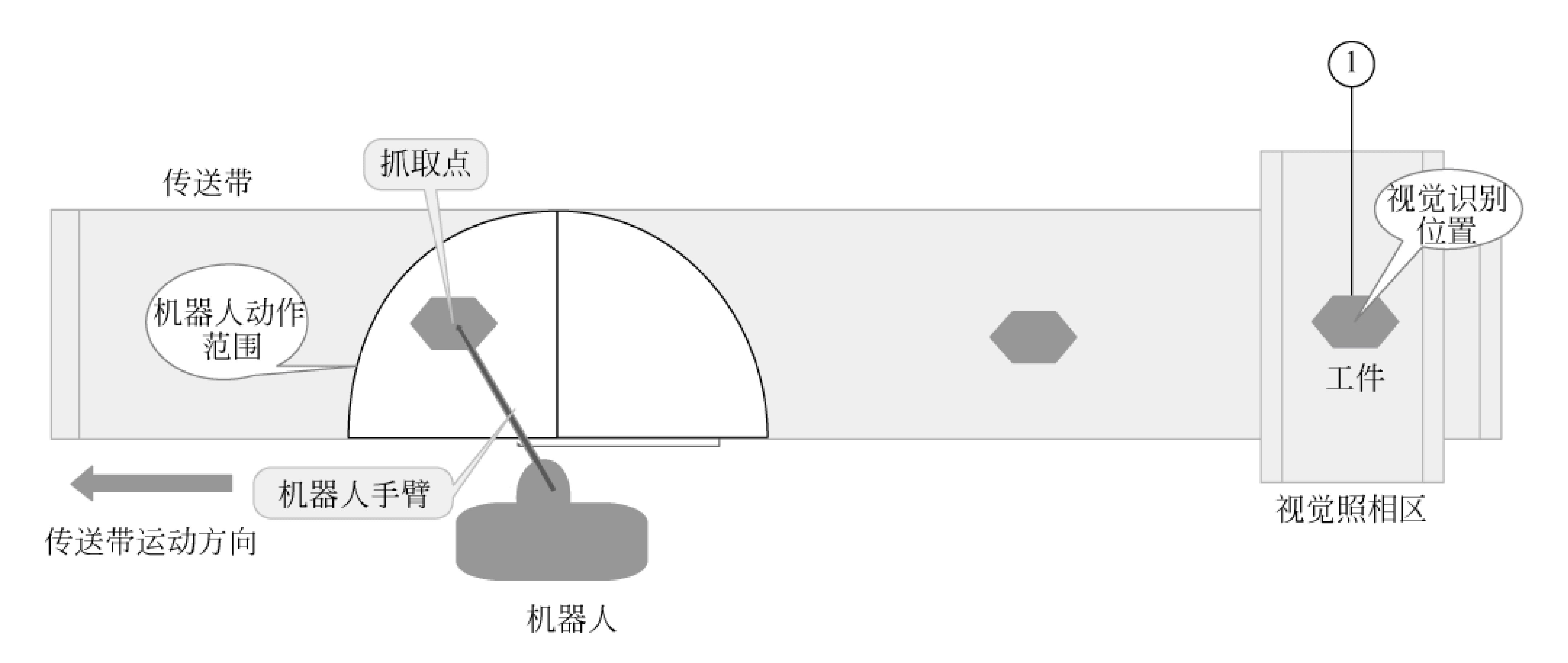


图 29-25　C 程序执行示意图

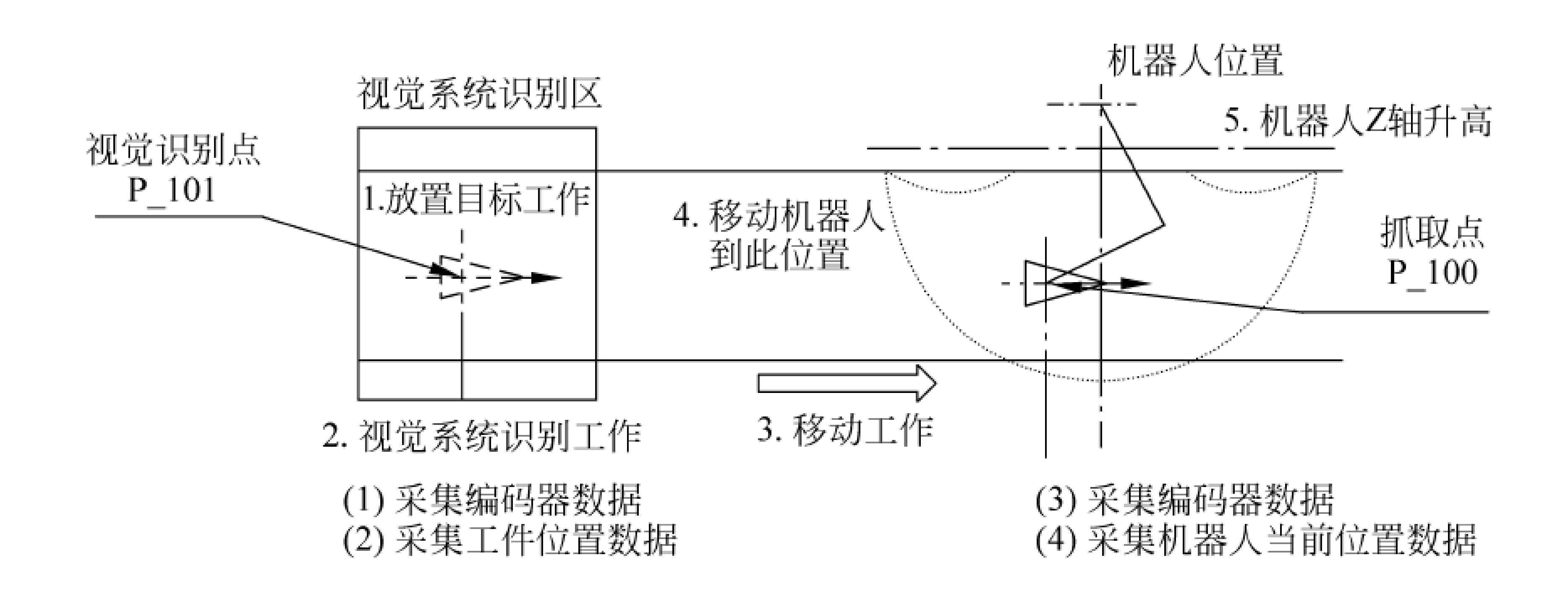


图 29-26　C 程序执行参考图

⑧ 如果是“面成像照相视觉系统”，可以在静态下拍照，则放置“标记工件”在照相区。视觉系统拍照完毕，发出信号给机器人，机器人获取相关信息。在 RT 软件上可监视编码器变量值（启动传送带，将工件移动进入追踪区）。

⑨ 如果是“线扫描成像视觉系统”，则必须在动态下拍照，所以必须把标记工件放置在传送带上，使“标记工件”通过视觉区一直运行到“追踪区”。

⑩ 在“追踪区”中部使传送带停止。

⑪ C 程序停止在 HLT 步。

⑫ 选择“手动模式"。

⑬ 使用机器人示教功能，标定机器人抓取点 PGT。

⑭ 选择“自动模式”，启动 C 程序直到执行到 END 步。

⑮ 标定出“待避点 P1”，“下料摆放点 PPT”。

⑯ C 程序执行完毕。

（2）C 程序执行完成后获得的全局变量如下。

① 编码器移动量：M_101＃。

② 视觉识别点：P_101。

③ 工件抓取点：P_100。

这些变量可以在 RT 软件中监视。

3. 操作后的检查确认

使用 RT 软件检查下列变量(根据工件类型数分别标注)。

(1) M_101()——编码器数据差。

(2) P_102()——变量 PRM1 中的数据(工件类型数/编码器编号)。

(3) P_103()——变量 PRM2 中的数据(图像识别区大小/工件尺寸)。

(4) C_100$()——通信口序号。

(5) C_101$()——视觉程序名称。

(6) C_102$()——识别工件数量。

(7) C_103$()——识别信息启动单元格。

(8) C_104$()——识别信息结束单元格。

4. 实用 C 程序

```
'——将"工件类型数量"设置为变量 PRM1 的 X 坐标。
'——将"编码器编号"设置为变量 PRM1 的 Y 坐标。
'——观察视觉传感器的实时画面,将移动方向的视界长度设置为变量 PRM2 的 X 坐标。
'——将工件的长度设置为变量 PRM2 的 Y 坐标值。
'——打开端口将 COM 的端口号输入至下行中"CCOM$ ="处。
  CCOM$ = "COM3:" '——设定通信端口号。
'——将视觉程序名称输入至下行中"CPRG$ ="处。
  CPRG$ = "TRK.JOB"   '——设定视觉程序名。
'——将工件放置在视觉传感器可以识别的位置。
'——将视觉传感器设置为"在线"。
'——自动运行 C 程序,程序停止的话,使用 T/B 再启动 C1 程序。
    MWKNO = PRM1.X   '——获取工件类型数量。
    MENCNO = PRM1.Y   '——获取编码器编号。
'——与视觉传感器端口连接。
   NVClose '——关闭端口。
    NVOpen CCOM$ As #1 '——开启端口指令。
    Wait M_NvOpen(1) = 1   '——等待端口打开。
    NVLoad #1, CPRG$   '——加载视觉程序。
    NVTrg #1,5,MTR1#,MTR2#   '——摄像要求 + 获取编码器值。
    EBRead #1,"",MNUM,PVS1,PVS2,PVS3,PVS4 '——分别获取各个识别工件的数据。
open "COM2:" as #1'——打开通信端口 2。
 wait m_open(1) = 1'——等待通信端口打开。
print #1,"SEO"'——输出字符串"SEO"。
input #1,mt1,mx1,my1,mc1'——读取数据。
pvs0.x = mx1'——赋值。
pvs0.y = my1'——赋值。
 'pvs0.c = mc1'——赋值。
 PVS0 = PVS1'——赋值。
 PVS1 = PVSCal(1,PVS0.X,PVS0.Y,PVS0.C)'——标定指令。
 P_101(MWKNO) = PVS1 '——获取"识别位置"工件的数据(将视觉系统识别的数据设置为一个全局
```

```
  变量)。
 ME1# = M_Enc(MENCNO) '——获取编码器数据 1。
NVClose #1; '——(以上程序读出了视觉系统的信息并进行了标定。也读出了编码器的数值。)
Hlt'——暂停。
'——移动传送带将工件移动至机器人动作范围内(将工件所在位置定为抓取点)。
 '—— 将机器人移动至"抓取点"。
   ME2# = M_Enc(MENCNO) '——获取编码器数据 2。
   P_100(MWKNO) = P_Fbc(1) '——获取机器人抓取点位置。
'——执行程序直至 END。
   MED# = ME2# - ME1#    '——计算编码器的移动量。
If MED#>800000000.0# Then MED# = MED# - 1000000000.0#
If MED#<-800000000.0# Then MED# = MED# + 1000000000.0#
 M_101#(MWKNO) = MED#   '——编码器移动量(设置为全局变量)。
   P_102(MWKNO) = PRM1  '——编码器编号(设置为全局变量)。
   P_103(MWKNO) = PRM2  '——视觉界面尺寸与工件尺寸(设置为全局变量)。
  C_100$(MWKNO) = CCOM$    '——COM 编号。
   C_101$(MWKNO) = CPRG$    '——视觉传感器程序名。
Hlt'——暂停。
End'——主程序结束。
'——(同时设置"待避点-原点 P1"和"下料摆放点 PPT")
'——本程序为获得视觉传感器识别的工件位置与机器人抓取位置的关系。
 '—— 运行前必须设置的变量。
PRM1 = (+1.000,+1.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
```

29.6 1#程序——自动运行程序

本节解释运行1#程序的操作流程。

1#程序在传送带追踪和视觉追踪中都是必要的。1#程序指令机器人追踪并抓取工件。工件的位置或是由光电开关识别或是由视觉系统识别。

29.6.1 示教

对“原点”和“下料摆放点”的示教操作流程如下。

(1) 用示教单元 TB 打开 1 号程序。

(2) 打开“位置数据编辑”屏幕。

(3) 当系统启动后,设置 P1 为原点。

(4) 移动机器人到原点位置并获取该位置,将其设置为 P1。

(5) 设置 PPT 为下料摆放点位置(该位置是工件最终被放置的位置)。

(6) 移动机器人到搬运终点位置并获取该位置数据设置为 PPT。

29.6.2 设置调节变量

1. 变量的定义及设置

本节解释在1#程序中的变量以及如何设置调节变量,有关的变量如表 29-3 所示。

表 29-3　变量及设置

变量名称	解释	设置样例
PWK “工件类型数量”	设置“工件类型数量”，即在一个追踪项目中，有几种类型的工件。 X=工件类型数量(1～10)	如果设置“工件类型数量”=1，则 (X，Y，Z，A，B，C)=(+1，+0，+0，+0，+0，+0)
PRI	1＃程序和CM1程序可以被同时执行，1＃程序控制机器人运动，CM1程序获取传感器或视觉识别数据。PRI用于指定程序的执行优先顺序，并不是在同一时间执行同一数量的程序，而是交替执行。 X=1＃程序执行的行数(1～31) Y=CM1程序执行的行数(1～31)	如果设置1＃程序运行1行而CM1程序运行10行，则(X，Y，Z，A，B，C)=(+1，+10，+0，+0，+0，+0)
PUP1 工作高度(抓取工件过程)	在抓取工件的操作中，设置机器人工作高度(mm)。 X=机器人“等待点”高度(mm) Y=抓取点的“近点”高度(在抓取之前) Z=抓取点的“近点”高度(在抓取之后) 由于Y和Z表示的是在TOOL坐标系中Z轴的位置，所以这些信号变量取决于机器人模式。	如果等待点高度=50mm， 抓取高度(抓取前)=－50mm 抓取高度(抓取后)=－50mm 则： (X，Y，Z，A，B，C)=(+50，－50，－50，+0，+0，+0)
PUP2——工作高度(释放工件过程)	在放置工件的操作中，设置机器人工作高度。高度是指释放工件的高度(mm)。 Y=工件释放位置的高度(在释放之前) Z=工件释放位置的高度(在释放之后) 由于Y和Z表示的是在TOOL坐标系中Z轴的位置，所以这些信号变量取决于机器人模式	如果释放高度(释放前)=－50mm。 释放位置(释放后)=－50mm。 则： (X，Y，Z，A，B，C)=(+0，－50，－50，+0，+0，+0)
PAC1	设置抓取工件的加减速时间比率。 X=加速到工件位置的倍率(1%～100%) Y=减速到工件位置的倍率(1%～100%)	如果设置加减速比率=100%，则(X，Y，Z，A，B，C)=(+100，+100，+0，+0，+0，+0)
PAC2	设置抓取工件的加减速时间比率。 X=加速到工件位置的倍率(1%～100%) Y=减速到工件位置的倍率(1%～100%)	如果设置加速比率=10%，减速比率=20%，则(X，Y，Z，A，B，C)=(+10，+20，+0，+0，+0，+0)
PAC3	设置朝着工件前进时抓取工件的加减速时间比率。 X=加速到工件位置的倍率(1%～100%) Y=减速到工件位置的倍率(1%～100%)	如果设置加速比率=50%，减速比率=80%，则(X，Y，Z，A，B，C)=(+50，+80，+0，+0，+0，+0)
PAC11	设置运动到释放工件位置时的加减速时间比率。 X=加速到释放工件位置的倍率(1%～100%) Y=减速到释放工件位置的倍率(1%～100%)	如果设置加速比率=80%，减速比率=70%，则(X，Y，Z，A，B，C)=(+80，+70，+0，+0，+0，+0)

续表

<table>
<tr><th>变量名称</th><th>解释</th><th>设置样例</th></tr>
<tr><td>PAC12</td><td>设置运动到释放工件位置时的加减速时间比率。
X＝加速到释放工件位置的倍率(1%～100%)
Y＝减速到释放工件位置的倍率(1%～100%)</td><td>如果设置加速比率＝5%，减速比率＝10%，则(X，Y，Z，A，B，C)＝(＋5，＋10，＋0，＋0，＋0，＋0)</td></tr>
<tr><td>PAC13</td><td>当执行释放工件动作时，设置朝工件位置运动时的加减速时间比率
X＝加速到工件位置的倍率(1%～100%)
Y＝减速到工件位置的倍率(1%～100%)</td><td>如果设置加速比率＝100%，减速速比率＝100%，则(X，Y，Z，A，B，C)＝(＋100，＋100，＋0，＋0，＋0，＋0)</td></tr>
<tr><td>PDLY1</td><td>设置抓取吸附时间
X：抓取吸附时间</td><td>如果设置吸附时间为为0.5秒，则
(X，Y，Z，A，B，C)＝(＋0.5，＋0，＋0，＋0，＋0，＋0)</td></tr>
<tr><td>PDLY2</td><td>设置释放时间
X：释放时间</td><td>如果设置释放时间为为0.3秒，则
(X，Y，Z，A，B，C)＝(＋0.3，＋0，＋0，＋0，＋0，＋0)</td></tr>
<tr><td>POFSET</td><td>当抓取位置改变后，其差值可以被设置。用本参数设置校正值。本参数是调整“抓取点”的重要参数</td><td></td></tr>
<tr><td>PTN
设置机器人和传送带以及工件移动方向的关系。简称机带关系</td><td>X值(1～6)<table><tr><th>设置值</th><th>传送带位置</th><th>传送带方向</th></tr><tr><td>1</td><td>前</td><td>从右到左</td></tr><tr><td>2</td><td>前</td><td>从左到右</td></tr><tr><td>3</td><td>左</td><td>从右到左</td></tr><tr><td>4</td><td>左</td><td>从前到后</td></tr><tr><td>5</td><td>右</td><td>从右到左</td></tr><tr><td>6</td><td>右</td><td>从左到右</td></tr></table></td><td>如果传送带在机器人前方，工件移动方向从右到左，则设置(X，Y，Z，A，B，C)＝(＋1，＋0，＋0，＋0，＋0，＋0)</td></tr>
<tr><td>PRNG</td><td>设置追踪区范围(在该范围内机器人能够追踪工件)
X＝开始位置
Y＝结束位置
Z＝临界位置</td><td>PRNG 与 PTN 的关系参见图 29-27～图 29-30</td></tr>
</table>

2. 追踪区范围及“机带关系”设置

(1) 传送带在机器人前方，工件移动方向为从右向左，则设置 PTN 的 X 坐标＝1；追踪区范围 PRNG：(X，Y，Z)＝(＋500，＋300，＋400)，如图 29-27 所示。

(2) 传送带在机器人前方，工件移动方向为从左向右，则设置 PTN 的 X 坐标＝2。追踪区范围 PRNG：(X，Y，Z)＝(＋300，＋100，＋200)，如图 29-28 所示。

(3) 传送带在机器人左侧，工件移动方向为从前向后，则设置 PTN 的 X 坐标＝4；追踪区范围 PRNG：(X，Y，Z)＝(＋400，＋200，＋300)，如图 29-29 所示。

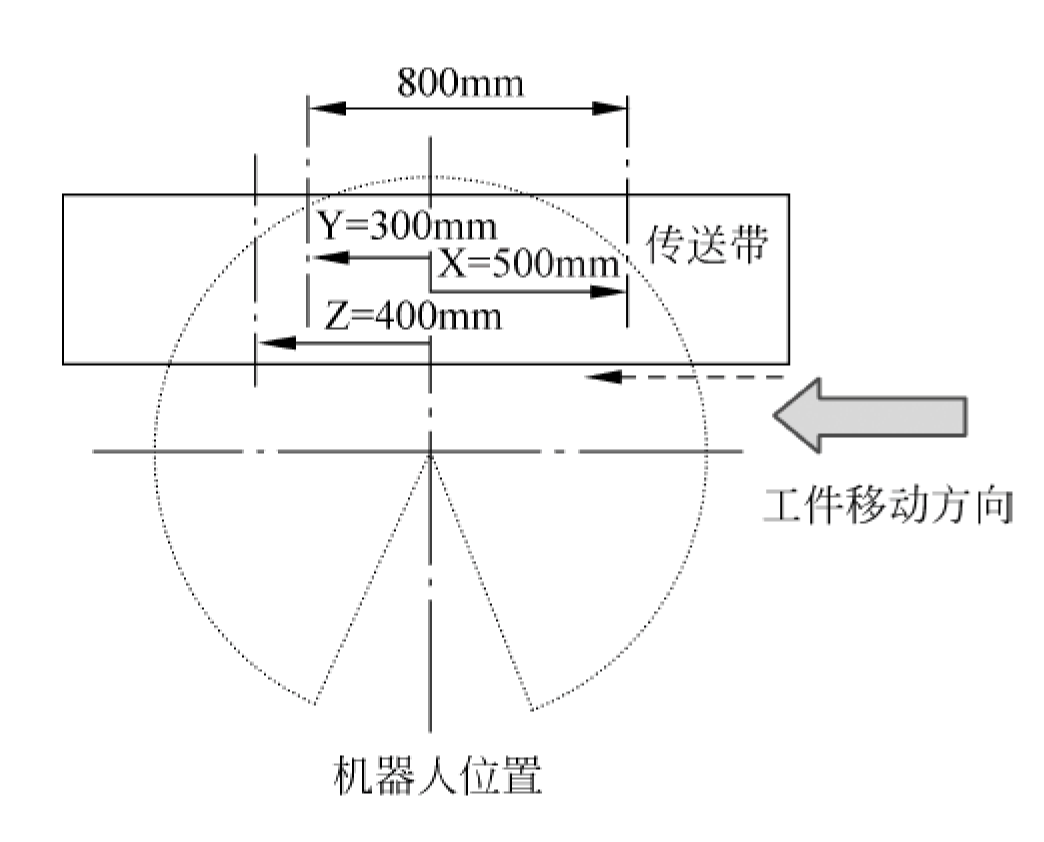


图 29-27 机器人追踪区域及机带关系布置图(PTN 的 X 坐标=1)

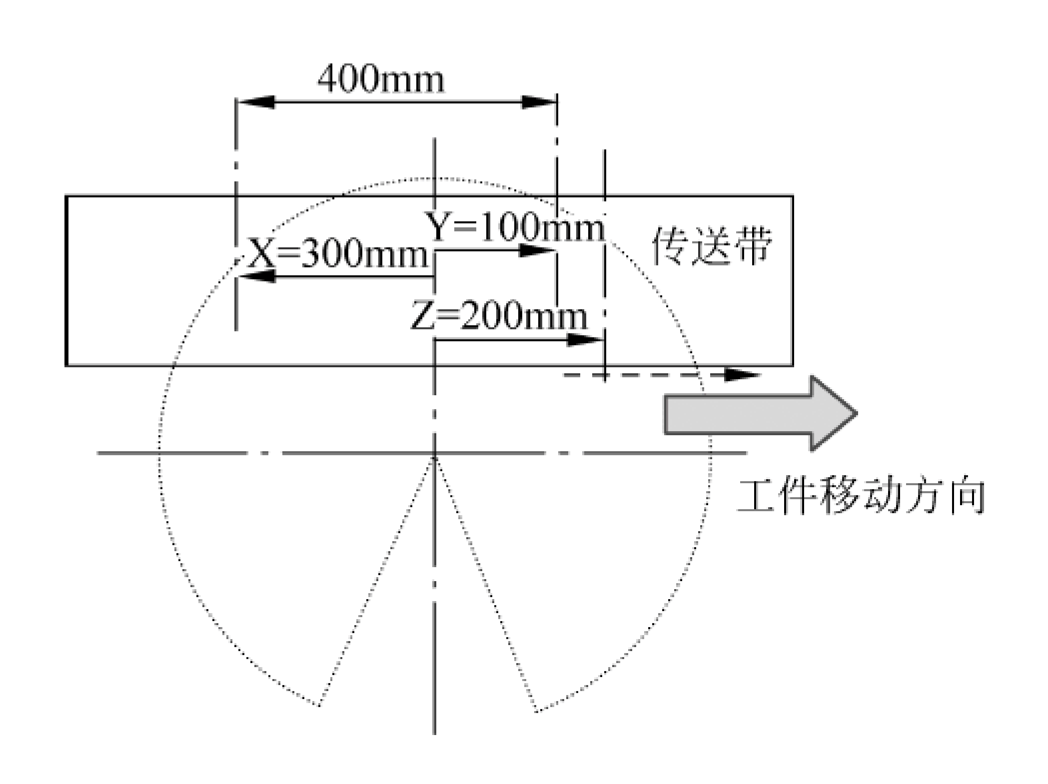


图 29-28 机器人追踪区域及机带关系布置图(PTN 的 X 坐标=2)

(4) 传送带在机器人右侧,工件移动方向为从后向前,则设置 PTN 的 X 坐标=5;追踪区范围 PRNG:(X,Y,Z)=(+500,+300,+400),如图 29-30 所示。

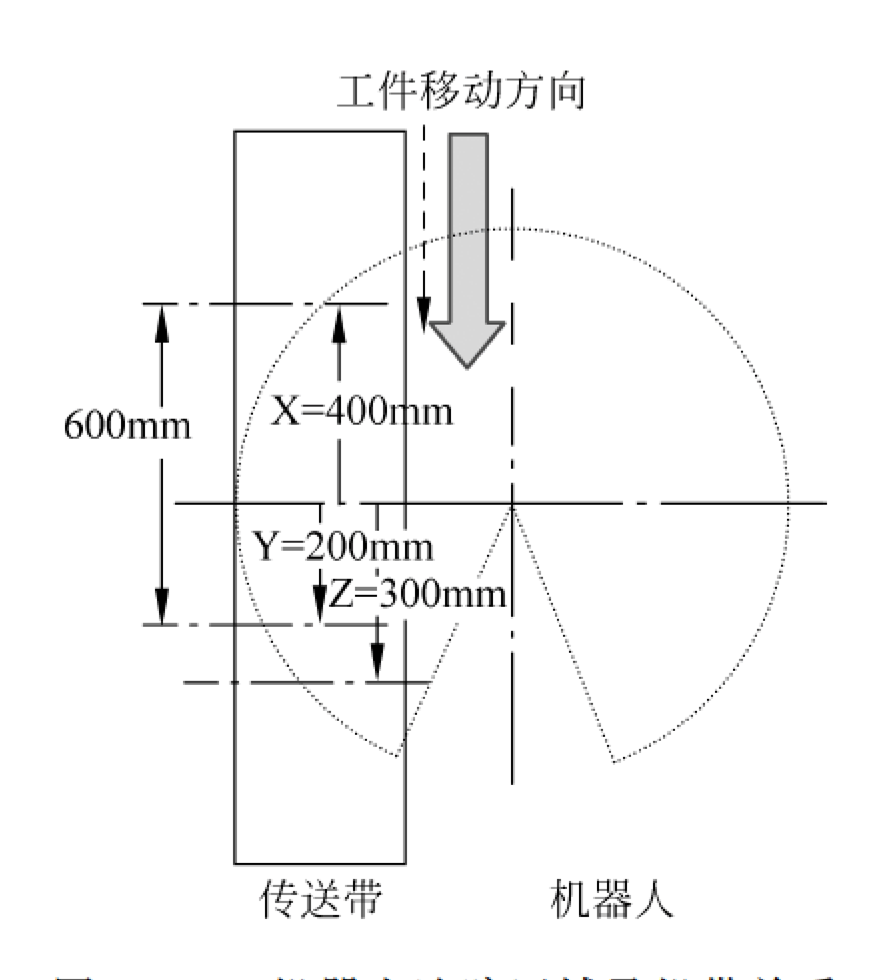


图 29-29 机器人追踪区域及机带关系布置图(PTN 的 X 坐标=4)

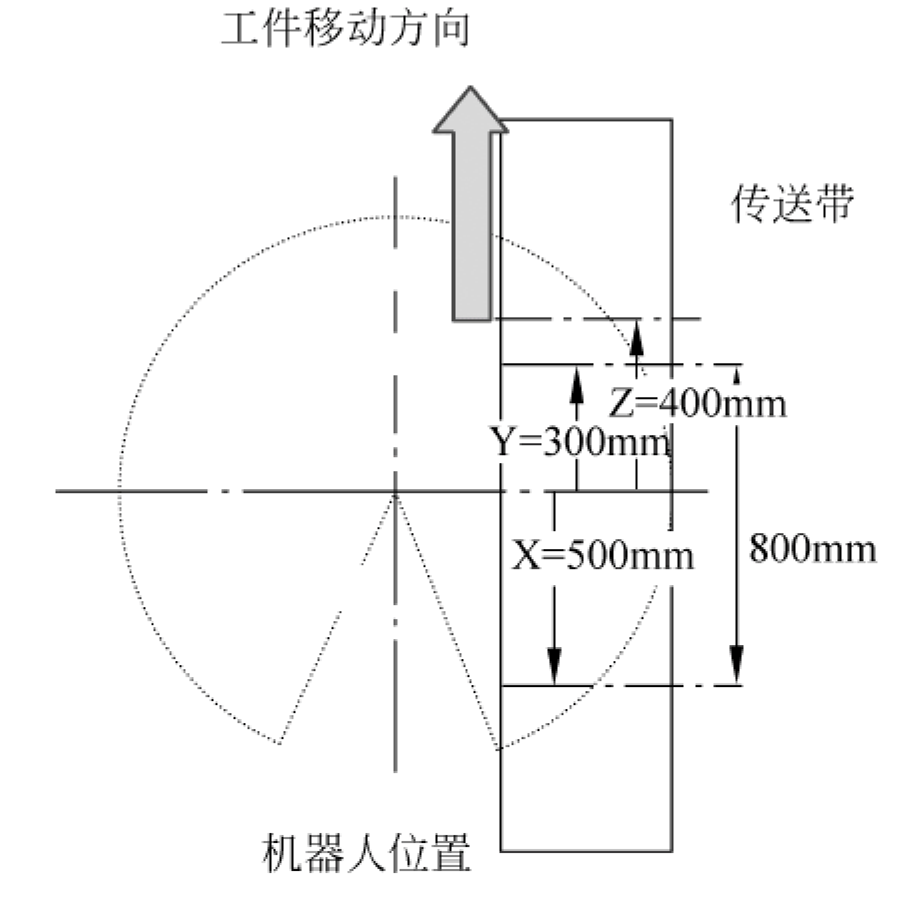


图 29-30 机器人追踪区域及机带关系布置图(PTN 的 X 坐标=5)

29.6.3 1#程序流程图

1#程序是自动运行程序。以主程序为框架,包含若干子程序。主程序结构流程如图 29-31 所示,其子程序如表 29-4 所示。

表 29-4 子程序汇总表

序　　号	子程序名称	功　　能
1	回原点子程序 * S90HOME	回原点
2	初始化子程序 S10INIT	进行初始化
3	抓取被追踪工件 S20TRGET	抓取被追踪工件
4	搬运放置工件 S30WKPUT	将工件放置在最终位置

1. 1＃主程序流程图

1＃主程序流程图如图 29-31 所示。

2. 回原点子程序流程图

回原点子程序流程图如图 29-32 所示。

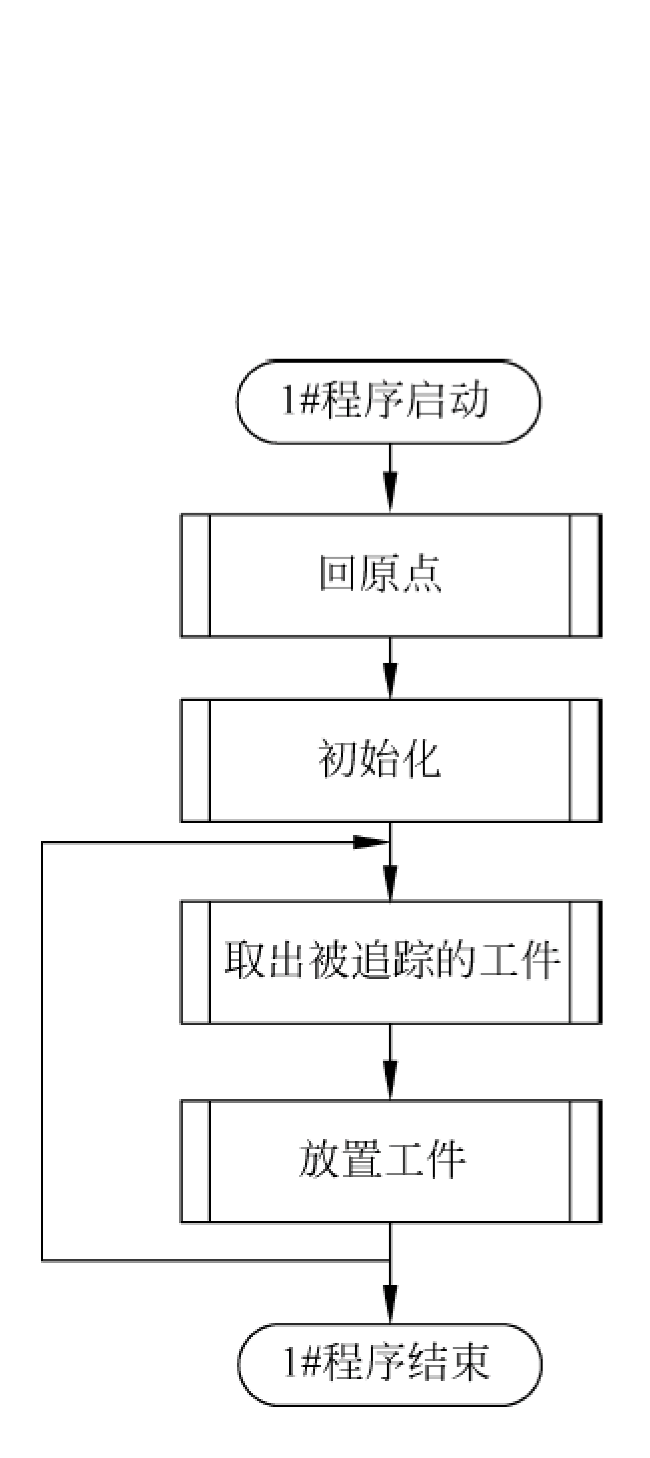


图 29-31　1＃程序流程图

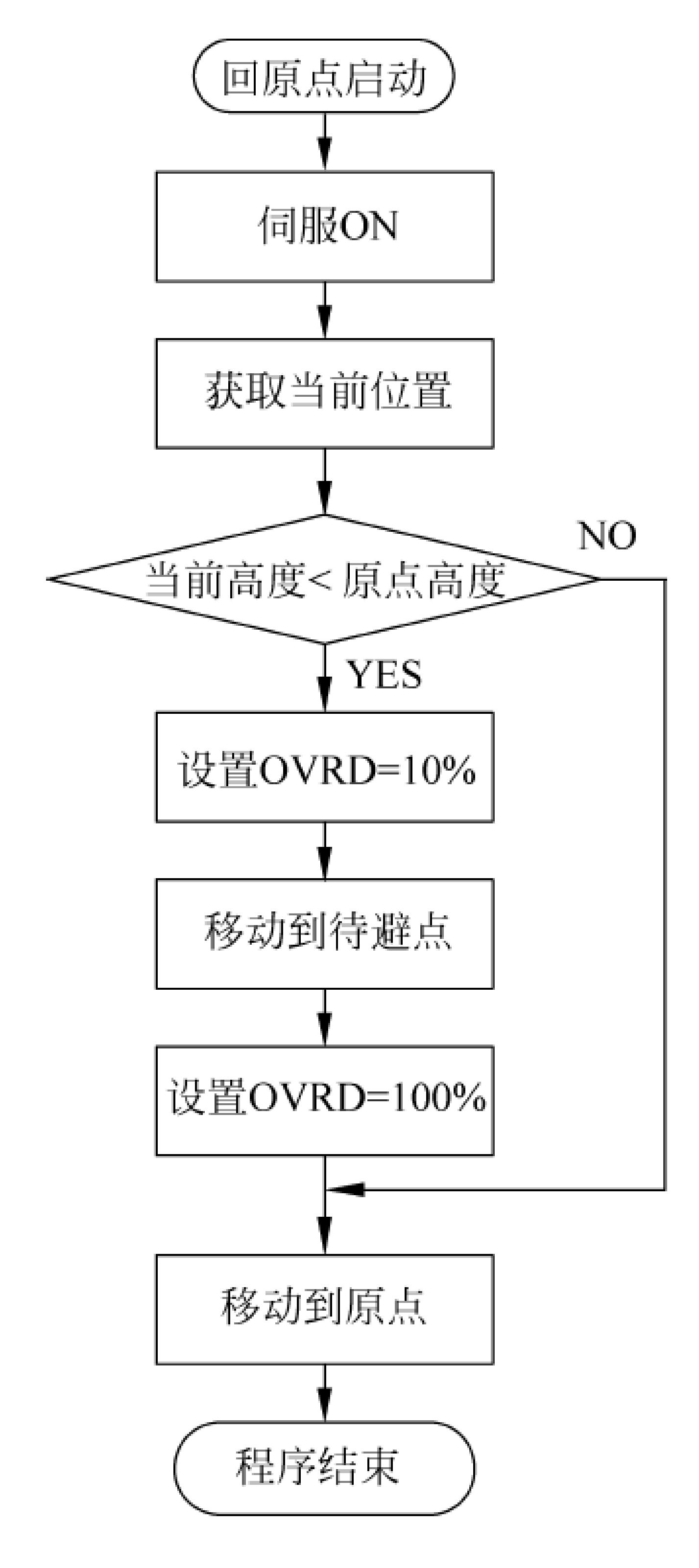


图 29-32　回原点程序流程图

回原点程序以 P1 点为原点(P1 点又作为“待避点”)。

(1) 判断机器人当前位置的“Z 轴坐标高度”是否小于 P1 点的“Z 轴坐标高度”?

(2) 如果小于,就直接升高当前点到 P1 点的“Z 轴坐标高度”。

(3) 如果不小于,就直接指令回到 P1 点。

3. 初始化子程序流程图

初始化子程序流程图如图 29-33 所示。

在“初始化程序”中,有一步很重要,就是“清除追踪缓存区内的数据”。因为每一次的工件视觉识别数据都被写入“追踪缓存区”,而 1＃程序是反复循环执行的,所以每执行完一次 1＃程序后,再执行新的程序前,要将追踪区内的数据清除,保证“追踪缓存区”内的数据为最新数据。

4. 搬运放置工件子程序流程图

搬运放置工件子程序流程图如图 29-34 所示。

5. 抓取被追踪的工件子程序流程图

抓取工件程序流程图如图 29-35 所示。

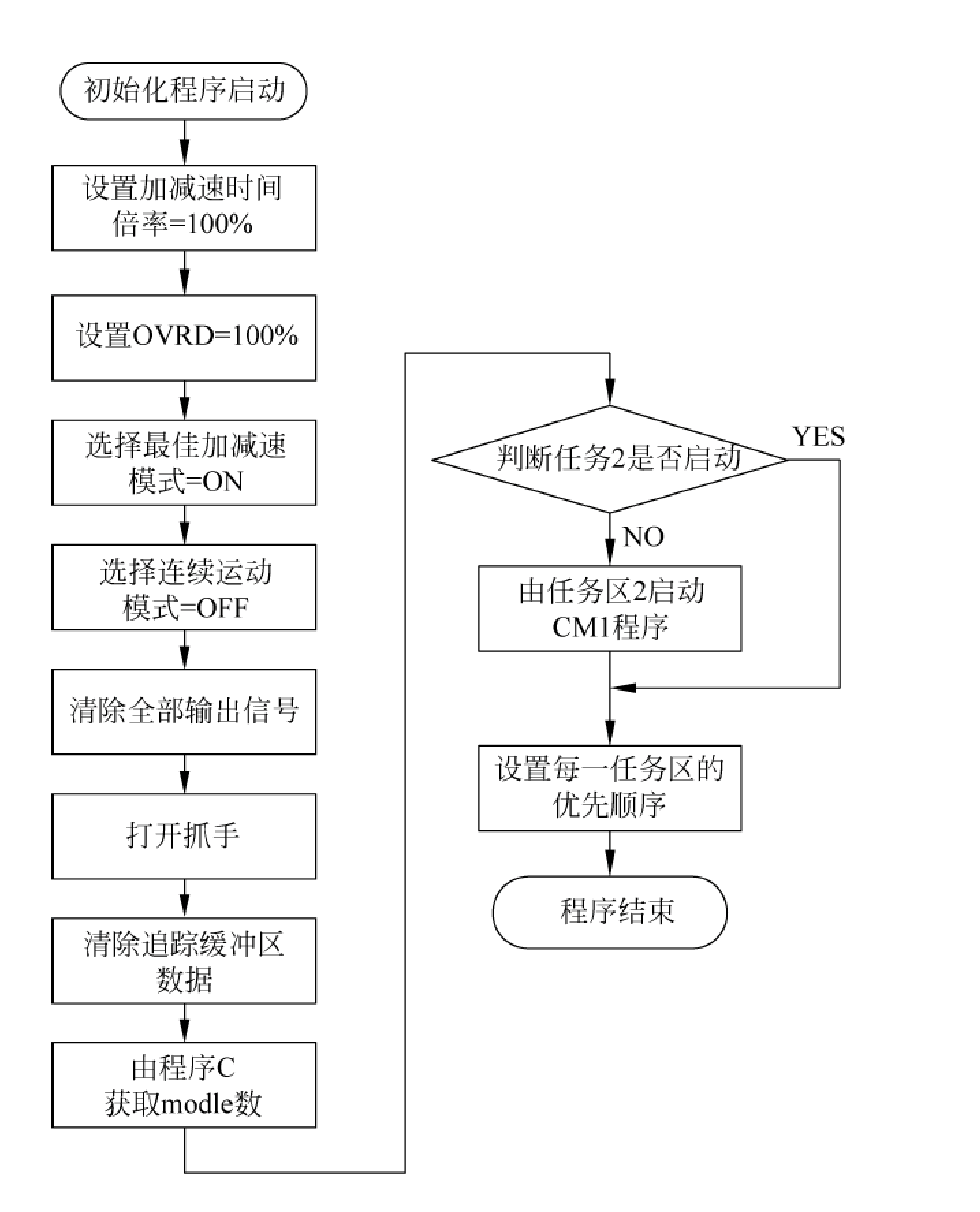


图 29-33　初始化程序流程图

图 29-34　搬运放置工件程序流程图

(1) 追踪动作各点位，如图 29-36 所示。

P500——待避点/原点(由操作者设定)。

P510——等待点(由 1＃程序中计算确定)。

P520——追踪启动点(由 1＃程序中计算确定)。

P530——抓取点(在 C 程序中设置)。

P540——下料摆放点(由操作者示教设定)。

(2) 对各位置点的详细说明如下。

① P500——待避点(由操作者设定)，也称为“机器人原点”，程序启动后及追踪全部动作完成后回到的“位置点”。

② P510——等待点(由 1＃程序计算确定)，当工件在“等待区”时的机器人位置点。“等待点”的意义就是指令机器人到达一个最佳位置，等待工件进入追踪区。在 1＃程序中，以“待避点 P1”为基准，在 X,Z,C 三个方向进行调整。

“等待点”一般设置在 X 方向与“当前工件坐标”相同，这是为了减少追踪时的行程。在 1＃程序中为 PWAIT 点。

③ P520——追踪启动点。追踪启动点是进入追踪模式后的追踪起点(在 1＃程序中由 TRBASE 指令设置)。

④ P530——抓料点。这是最重要的一点。在 C 程序中具体示教确定(设定为全局变量

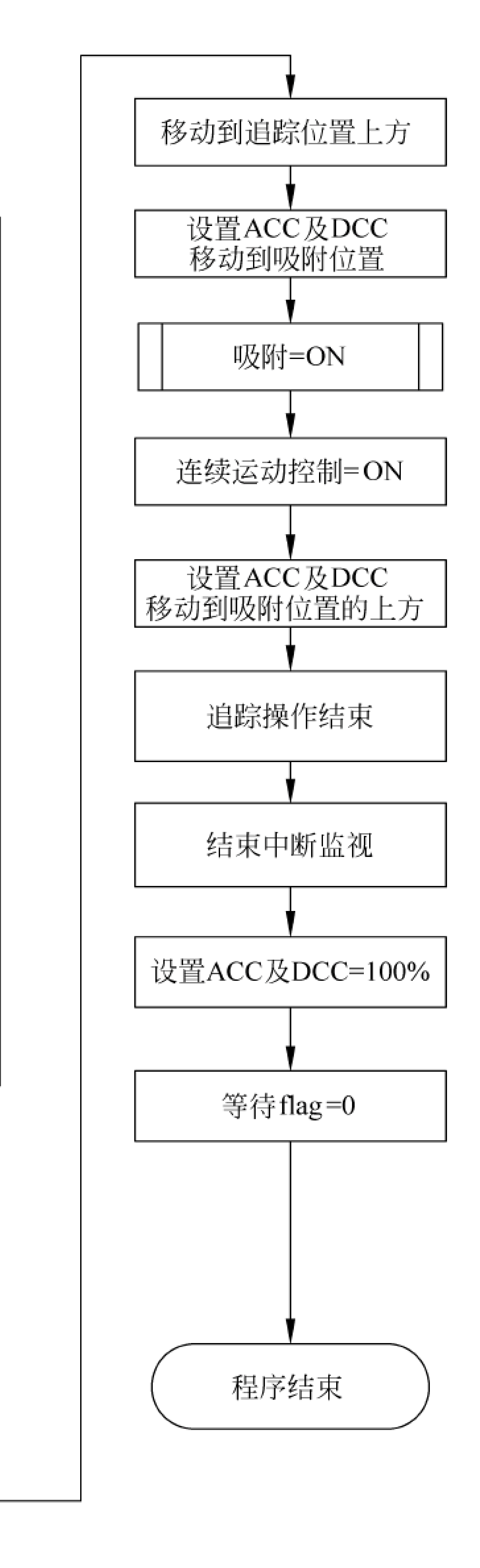


图 29-35　抓取工件程序流程图

P_100)。

P530 抓料点的设置应该在靠近追踪区结束位置,这样设置是为了满足工件相隔距离较小的场合,在 1# 程序中为 PGT 点。

⑤ P540——下料摆放点(由操作者示教设定)。下料摆放点根据实际工况要求确定,在 1# 程序中为 PPT 点。

6. 传送数据子程序流程图

传送数据子程序流程图如图 29-37 所示。

7. 1# 程序的运动流程及各点位的关系

1# 程序的运动流程及各点位的关系如图 29-38 所示。运动流程如下。

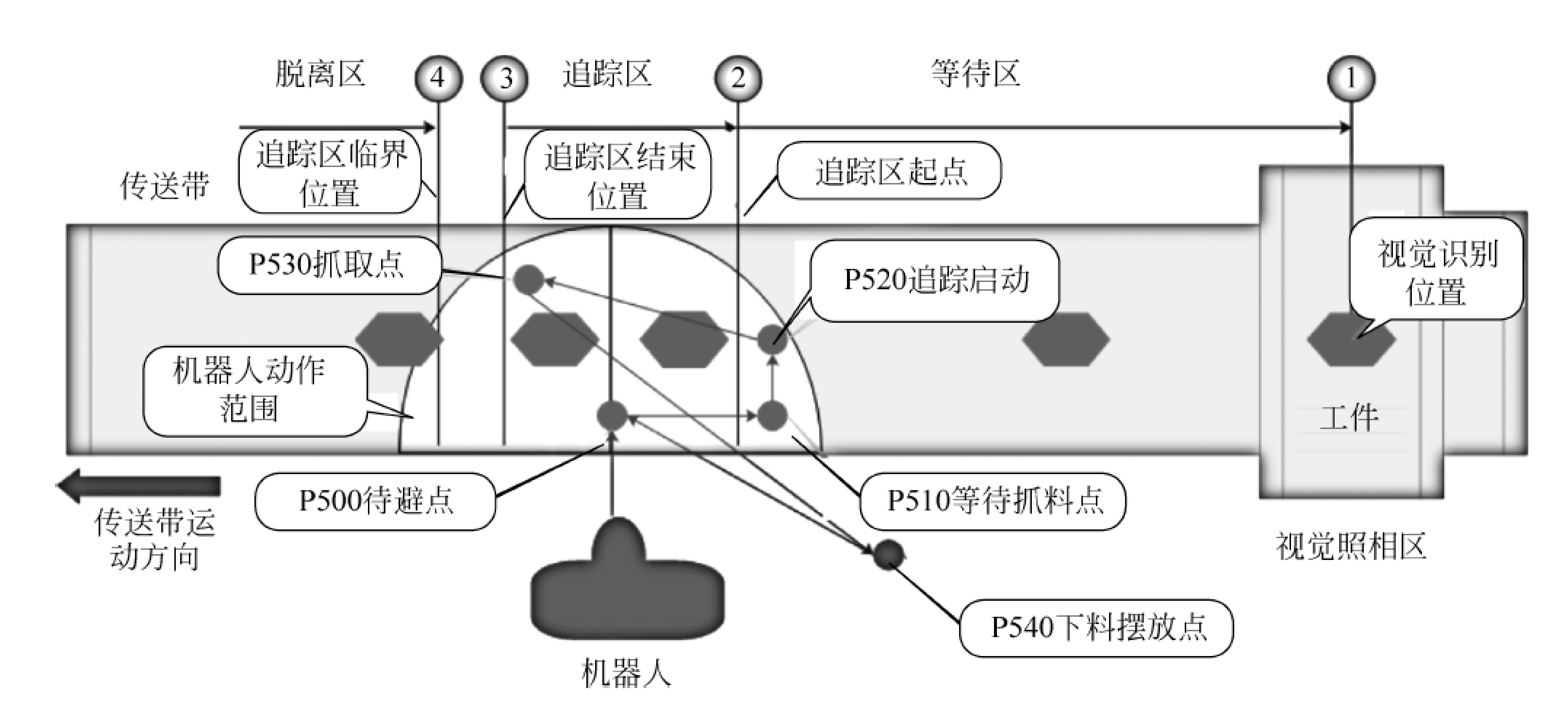


图 29-36 追踪动作各点位示意图

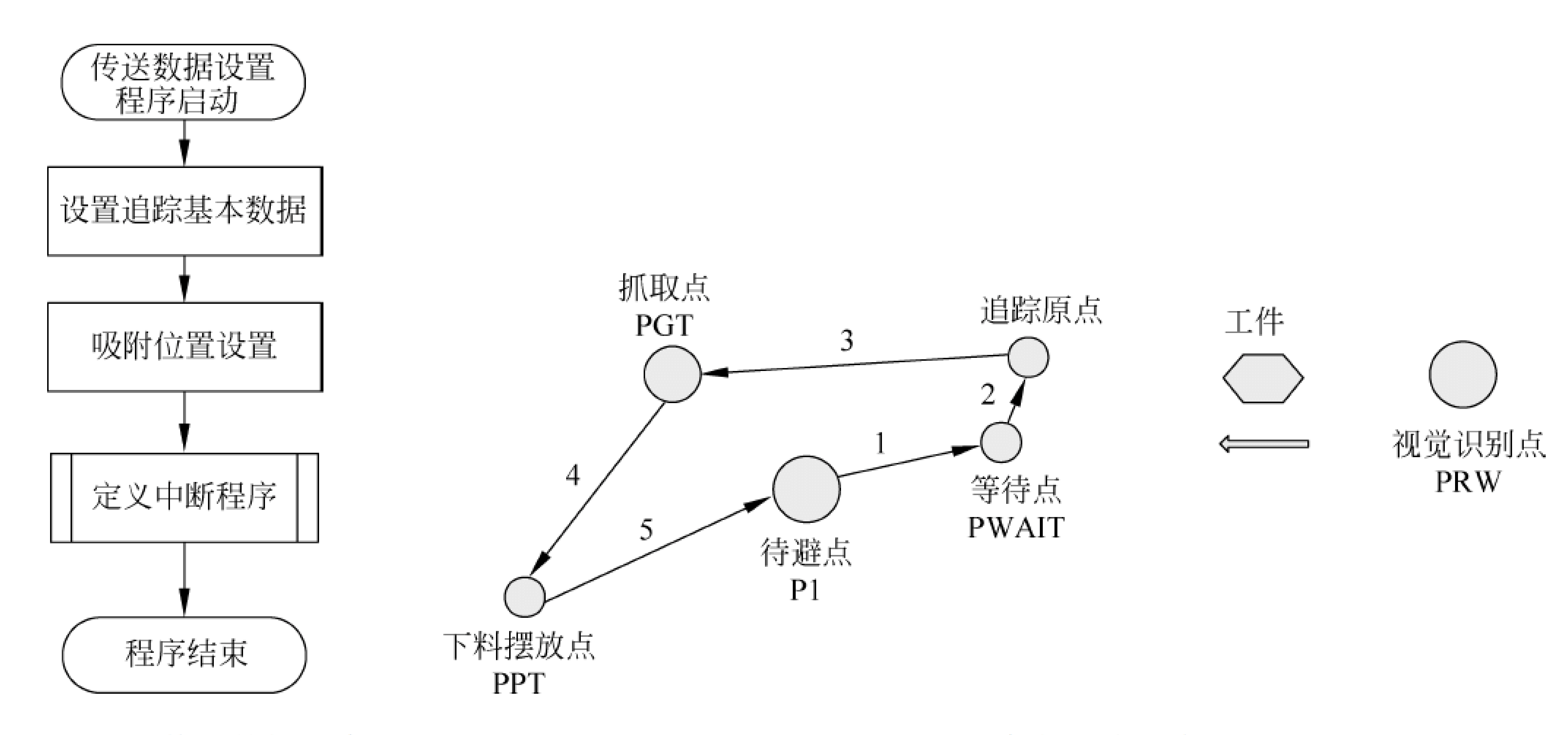


图 29-37 传送数据程序流程图

图 29-38 1#程序各点位示意图

(1) 程序启动,机器人运动到待避点 P1。

(2) 在追踪缓存区判断是否有视觉数据(视觉数据由 CM1 程序写入),如果没有,则继续判断循环。

(3) 如果有,机器人运动到等待点 PWAIT。

(4) 做当前工件位置判断。

① 如果在“等待区”,机器人就一直停在等待点 PWAIT。

② 如果进入“追踪区”,机器人就启动追踪模式。

(5) 进入追踪模式后,移动到追踪原点 PTBASE。在 1#程序中,PTBASE=P_100,即追踪原点=抓取点。

(6) 移动到“抓取点上方”—— Mov PGT,PUP1.Y Type0,0。

(7) 移动到“抓取点”—— Mov PGTType0,0。

(8) 移动到“抓取点上方”——Mov PGT,PUP1.Z Type0,0。

(9) 移动到“下料摆放点 PPT”。

29.6.4 实用1#程序

```
'###主程序 ###
 *S00MAIN  '程序分支标记
GoSub *S90HOME '——调回原点子程序
GoSub *S10INIT '—— 调初始化子程序
 *LOOP'—— 循环标志
GoSub *S20TRGET '——调用抓取工件子程序
GoSub *S30WKPUT '——调放置工件子程序
GoTo *LOOP'——跳转到*LOOP行(反复执行以上两个程序)
End'——主程序结束。
'###初始化子程序###
 *S10INIT
'/// Speed related ///  设置速度倍率相关程序
Accel 100,100 '——设置加减速时间。
Ovrd 100 '——设置速度倍率。
LOADSET 1,1 '——设置抓手工作条件
Oadl On'——启动最佳工作模式。
Cnt 0'——连续轨迹运行无效。
Clr 1'—— 输出点复位
HOpen 1'——抓手张开。
'///初始值设置///
TrClr 1 '——对追踪缓存区 1 清零(重要)
MWAIT1 = 0 '——  对"工件到达标志"执行清零。
 '///多任务启动(启动任务区 2 内的程序"CM1")///
M_09# = PWK.X '——设置工件类型数量。PWK.X是专门规定用于设置工件类型数的变量。PWK.X
为位置变量形式,所以必须在位置变量区域设置。参见程序最后部分。
If M_Run(2) = 0 Then '——M_Run为状态变量,表示任务区内程序执行状态。M_Run(2) = 0 表示
任务区 2 内的程序处于停止状态。
 XRun 2,"CM1",1 '——指令任务区 2 内的 CM1 程序做单次运行。
Wait M_RUN(2) = 1'—— 等待任务区 2 内的程序启动
(以上是启动执行任务区 2 内的 CM1 程序。)
EndIf'——判断语句结束。
Priority PRI.X,1  '——执行在任务区 1 内的程序,行数由变量 PRI 的 X 坐标值确定(变量 PRI
需要预先设置)
Priority PRI.Y,2'——执行在任务区 2 内的程序,行数由变量 PRI 的 Y 坐标值确定(变量 PRI 需
要预先设置)
Priority——多任务工作时,指定各任务区程序的执行行数
Return'——子程序结束。
'### 抓取被追踪工件子程序
  *S20TRGET'——程序分支标志
'///追踪缓存区数据检查///
  *LBFCHK'——程序分支标志(检查追踪缓存区数据)
If M_Trbfct(1)>=1 Then GoTo *LREAD '——  判断"追踪缓存区"内是否写入视觉识别数据,如
果否,就回原点等待。如果是,就跳转到*LREAD。
M_Trbfct——状态变量。表示在"追踪缓存区内"的数据内容。
Mov P1 Type 0,0 '——(如果 NO)移动到 P1 点。P1 点是退避点位置。
MWAIT1 = 0  '—— 发出 MWAIT1 = 0 信号,MWAIT1 是自定义变量,表示机器人处于等待状态。
```

39 GoTo * LBFCHK '—— 返回 * LBFCHK 继续进行缓存区数据检查。

40 '///工件数据采集///

41 * LREAD'——(如果追踪缓存区内有写入的数据,就从 36 行跳转到本行)

42 TrRd PBPOS,MBENC#,MBWK%,1,MBENCNO% ' ——从"追踪缓存区"读数据。读出的数据分别存储。"位置数据"存储在 PBPOS、"编码器数值"存储在 MBENC#、"工件类型数量"存储在 MBWK%、缓存区序号 = 1,"编码器编号"存储在变量 MBENCNO%。

43 GoSub * S40DTSET '——调子程序 * S40DTSET。 * S40DTSET 子程序用于生成追踪起点和抓取点位置。

44 '///工件位置判断///

45 * LNEXT'——程序分支标志

46 PX50CUR = TrWcur(MBENCNO%,PBPOS,MBENC#)'——(获取当前工件位置数据。TrWcur 是函数,用于获得当前点的编码器数值和当前工件位置。由于工件在传送带上运行,所以 PX50CUR 是一个动态值。)

MBENCNO%——编码器编号;

PBPOS——视觉点位置;

MBENC#——编码器数值;

PX50CUR——是当前工件点的位置;

47 MX50ST = PRNG.X '—— 设置追踪区启动位置线;

48 MX50ED = PRNG.Y '——设置追踪区结束位置线;

49 MX50PAT = PTN.X '—— 设置机 - 带位置编号。PTN 用于表示传送带与机器人的位置关系。

50 GoSub * S50WKPOS '——调用确认工件位置程序。

'—— 以下是根据当前工件位置进行"等待""追踪""放弃"工作模式的选择。

51 If MY50STS = 3 Then GoTo * LBFCHK '—— MY50STS 是一个变量,MY50STS 是工件当前位置的判断结果,如果已经超出追踪区范围,就跳转到 * LBFCHK(在 35 行),读取下一工件信息。MY50STS = 3 表示工件已经超出机器人工作范围。

52 If MY50STS = 2 Then GoTo * LTRST '——如果 MY50STS = 2,就跳转到 * LTRST,启动追踪程序(工件进入追踪区)。

53 If MWAIT = 1 Then GoTo * LNEXT '——如果 MWAIT = 1,就跳转到 * LNEXT,再进行工件位置判断。

54 '///到等待点 PWAIT ///

55 PWAIT = P1 '——(P1: "待避点")把 PWAIT 点设置成为"待避点"。

56 Select PTN.X '——根据 PTN(机器人与传送带位置关系)选择程序流程;

57 Case 1 TO 2 ' ——如果 PTN.X = 1～2,则:

58 PWAIT.X = PX50CUR.X '—— 赋值。将当前工件点位置的 X 值赋予 PWAIT.X。

59 Case 3 TO 6 '—— 如果 PTN.X = 3～6,则:

60 PWAIT.Y = PX50CUR.Y '—— 赋值。将工件被检测点位置的 Y 值赋予 PWAIT.Y。

61 End Select'——选择语句结束。

62 PWAIT.Z = PX50CUR.Z + PUP1.X'——

PUP1.X: 待机高度(待机点的高度 = 当前工件点高度 + 调整值)。

63 PWAIT.C = PX50CUR.C'—— 赋值。待机位置的 C 角度 = 当前工件位置 C 角度

(以上对待避点设置完毕)。

64 Mov PWAIT Type 0,0 '——移动到待机点 PWAIT。

65 MWAIT1 = 1 '——发出机器人到达"等待点"标志(表示机器人已经到达待机位置。等待工件进入追踪区。以上程序是设置一个"等待点",等待点 PWAIT 的 X、Y 值与检测工件点相同。Z 值比检测点高一个"调整量",C 值与检测点相同)。

66 GoTo * LNEXT'——跳转回"工件位置判断"程序行。

67 '///启动追踪操作///包含追踪启动指令,到达吸附点的动作。

68 * LTRST '——程序分支标志(追踪程序标志)

69 Accel PAC1.X,PAC1.Y '——加减速时间设置。

70 Cnt 1,0,0 '——连续轨迹

71 Act 1 = 1 '——中断程序 1 有效区间起点。

72 Trk On,PBPOS,MBENC#,PTBASE,MBENCNO% '——追踪启动(包括移动到追踪起点及追踪工件的动作)

73 Mov PGT,PUP1.Y Type 0,0'——移动到抓取位置 PGT 上方近点(PGT 由"抓取点 + 调节量"构成)。PUP1.Y 为抓取前位置(高度)

74 Accel PAC2.X,PAC2.Y '——加减速时间设置。

75 Mvs PGT '—— 移动到抓取位置。

76 HClose 1 '——抓手动作。

77 Dly PDLY1.X '——暂停,确认抓取。

PDLY1.X——预先设置的吸附时间。

78 Cnt 1'——连续路径

79 Accel PAC3.X,PAC3.Y'——加减速时间设置。

80 Mvs PGT,PUP1.Z '—— 移动到抓取(完成)位置点。PUP1.Z 为预设的抓取(完成)位置点。

81 Trk Off '—— 追踪结束。

82 Act 1 = 0 '——中断程序 1 的有效工作区间结束点。

83 Accel 100,100'——设置加减速时间。

84 MWAIT = 0 '—— 设置等待标志 = 0。

85 Return '——子程序返回。

86 '###工件摆放程序###

87 *S30WKPUT'——程序分支标志。

88 Accel PAC11.X,PAC11.Y '——设置加减速时间。

89 Mov PPT,PUP2.Y '——移动到摆放位置上部。PPT 为摆放位置。

90 Accel PAC12.X,PAC12.Y '——设置加减速时间。

91 Cnt 1,0,0'——连续轨迹运行。

92 Mvs PPT '——移动到放置位置。

93 HOpen 1 '——抓手 = OFF。

94 Dly PDLY2.X '—— 释放确认。

95 Cnt 1'——连续轨迹运行。

96 Accel PAC13.X,PAC13.Y '——设置加减速时间。

97 Mvs PPT,PUP2.Z '——移动到放置位置上点。

98 Accel 100,100 '——设置加减速时间。

99 Return '——子程序返回。

100 '——追踪数据设置(设置追踪起点、抓取点位置及调节量)。

101 *S40DTSET(生成追踪起点、抓取工作点)。

102 PTBASE = P_100(PWK.X) '——生成追踪起点(P_100——C 程序确定的抓取点)。

103 TrBase PTBASE,MBENCNO% '——设置追踪起点 (注意,这时的追踪起点为示教抓取点 P_100)。

104 PGT = PTBASE * POFSET '—— 设置抓取位置(对抓取点进行精度调节,注意是乘法运算)。

PTBASE——追踪起点;

POFSET——抓取点调节量(主要用于调整角度);

PGT——抓取位置;

105 GoSub *S46ACSET'——调用子程序。

106 Return'——子程序结束

107 '——###中断程序 1 ###

以下程序判断机器人的动作范围是否超过"追踪区临界距离"。如果超出范围,就结束追踪动作。

108 *S46ACSET'——程序分支标志

109 Select PTN.X '——根据机 - 带位置关系选择动作;

110 Case 1 '——如果 PTN.X = 1

111 MSTP1 = PRNG.Z '——设置"追踪临界值"。PRNG.Z——追踪临界值,超过该值就停止追踪。

112 Def Act 1,P_Fbc(1).Y > MSTP1 GoTo *S91STOP,S '——定义中断程序:如果机器人的当前位置(Y 坐标)大于"追踪临界值"就 GoTo *S91STOP。

113 Break'——结束选择语句"Select PTN.X"。

114 Case 2 '——如果 PTN.X = 2。

115 MSTP1 =- PRNG.Z '——设置"追踪临界值"为 - PRNG.Z。

116 Def Act 1,P_Fbc(1).Y < MSTP1 GoTo *S91STOP,S

```
'—— 如果机器人的当前位置(Y坐标)小于"追踪临界距离"就GoTo *S91STOP
Break'——结束选择语句"Select PTN.X"。
Case 3 '—— 如果PTN.X=3,
    Case 5 '——如果PTN.X=5
MSTP1=PRNG.Z '——设置"追踪临界值"为PRNG.Z。
Def Act 1,P_Fbc(1).X>MSTP1 GoTo *S91STOP,S
如果机器人的当前位置(X坐标)大于"追踪临界值"就GoTo *S91STOP,S
 Break'——结束选择语句"Select PTN.X"。
Case 4 '——如果PTN.X=4
 Case 6 '——如果PTN.X=6
 MSTP1=-PRNG.Z'——设置"追踪临界值"为-PRNG.Z。
Def Act 1,P_Fbc(1).X<MSTP1 GoTo *S91STOP,S
'——判断:如果机器人的当前位置(X坐标)小于"追踪临界值"就GoTo *S91STOP,S
Break'——结束选择语句"Select PTN.X"。
End Select'——结束选择语句"Select PTN.X"。
 Return'——子程序结束。
'以上程序用于判断机器人的动作范围是否超过"临界追踪值"。
'###确认工件位置程序###。
以下程序用于判断工件是否在工作区域,判断以后给出标志。
'PX50CUR'——当前工件位置(注意是工件位置而不是机器人位置)
'MX50ST——追踪区启动线
'MX50ED——追踪区结束线(MX50ED=PRNG.Y)
'MX50PAT=PTN.X.——表示机器人与传送带的位置关系。
'MY50STS——当前工件位置的判断结果。(MY50STS =1:工件在等待区;MY50STS= 2:工件进入
追踪区;MY50STS =3:工件已经超出追踪区。)
 *S50WKPOS'——程序分支标志
MY50STS=0 '——清除MY50STS原结果数据。
 Select MX50PAT '—— 根据机带位置关系设置追踪区范围及判断工件当前位置;
Case 1 '——如果PTN.X=1
 M50STT=-MX50ST '——(MX50ST=PRNG.X)MX50ST=追踪区起点线。
M50END=MX50ED'——(MX50ED=PRNG.Y)MX50ED=追踪区结束线。
 If Poscq(PX50CUR)=1 And PX50CUR.Y>=M50STT And PX50CUR.Y<=M50END Then
 '—— Poscq是检测"工件当前位置"是否在机器人工作范围的运算函数。本行指令是判断如
 果"工件当前位置"在机器人工作范围,而且PX50CUR.Y大于追踪区起点,PX50CUR.Y小于等于追
 踪区终点,则
 MY50STS=2 '——设置MY50STS=2 可进入追踪模式
Else '——否则执行下一行
If PX50CUR.Y<0 Then MY50STS=1 '——再判断。工件当前位置Y<0,则设置MY50STS=1进入等
待模式。
If PX50CUR.Y>M50END Then MY50STS=3 '——再判断。工件当前位置Y>M50END,则设置MY50STS
=3(工件已经超出追踪范围)。
If Poscq(PX50CUR)=0 And PX50CUR.Y>=M50STT And
PX50CUR.Y<=M50END Then
MY50STS=3 '——判断:如果工件当前位置超出机器人工作范围,也设置MY50STS=3。表示工件
已经超出了动作区域。
EndIf'——结束判断语句。
Break'——结束选择语句"Select MX50PAT"。
 Case 2 '——如果PTN.X=2
M50STT=MX50ST'——设置追踪区起点
M50END=-MX50ED '——设置追踪区终点。
 If Poscq(PX50CUR)=1 And PX50CUR.Y<=M50STT And PX50CUR.Y>=M50END Then
```

```
155 MY50STS = 2 '——可执行追踪。
156 Else '——否则不能执行追踪。
157 If PX50CUR.Y > 0 Then MY50STS = 1 '——等待。
158 If PX50CUR.Y < 0 Then MY50STS = 3 '——移动到下一工件。
159 If Poscq(PX50CUR) = 0 And PX50CUR.Y < = M50STT And PX50CUR.Y > = M50END Then
    MY50STS = 3 '——超出追踪区范围。
160 EndIf'——结束判断语句。
161 Break'——结束选择语句"Select MX50PAT"。
162  Case 3 '——如果 PTN.X = 3
163 Case 5 '——如果 PTN.X = 5
164 M50STT =- MX50ST '——设置追踪区起点。
    65   M50END = MX50ED'——设置追踪区终点。
166 If Poscq(PX50CUR) = 1 And PX50CUR.X > = M50STT And PX50CUR.X < = M50END Then
167 MY50STS = 2 '——可执行追踪
168 Else '——如果不能够执行追踪
169 If PX50CUR.X < 0 Then MY50STS = 1 '——等待。
170 If PX50CUR.X > 0 Then MY50STS = 3 '——移动到下一工件。
171 If Poscq(PX50CUR) = 0 And PX50CUR.X > = M50STT And PX50CUR.X < = M50END Then
MY50STS = 3 '——超出追踪区范围。
172  EndIf'——结束判断语句。
173 Break'——结束选择语句"Select MX50PAT"。
174 Case 4 '——如果 PTN.X = 4
175 Case 6 '——如果 PTN.X = 6
176 M50STT = MX50ST'——设置追踪区起点。
177 M50END =- MX50ED '——设置追踪区终点。
178  If Poscq(PX50CUR) = 1 And PX50CUR.X < = M50STT And PX50CUR.X > = M50END Then
179 MY50STS = 2 '——可执行追踪。
180 Else '——如果不能执行追踪。
181 If PX50CUR.X > 0 Then MY50STS = 1 '——等待。
182 If PX50CUR.X < 0 Then MY50STS = 3'——移动到下一工件。
183 If Poscq(PX50CUR) = 0 And PX50CUR.X < = M50STT And PX50CUR.X > = M50END Then
    MY50STS = 3 '——超出追踪区范围
184 EndIf'——结束判断语句。
185  Break'——结束选择语句"Select MX50PAT"。
186 End Select'——结束选择语句"Select MX50PAT"。
187 If MY50STS = 0 Then Error 9199'——报警而且应该修正程序
188  Return'——子程序结束
189 '—— 以上程序根据机器人与传送带的位置关系，判断当前工件是否在机器人的追踪范围之内。
    从而发出判断信号(等待、可追踪、工件超出追踪区)
    190 '——###原回点###(比较当前位置点是否低于"待机点高度(Z)"，如果低于"待机点
    高度(Z)"，则直接提升到"待机点高度(Z)"，如果大于"待机点高度(Z)"，则回到 P1 点。P1 点既是
    待避点，也是原点)
191  *S90HOME'——程序分支标志
192 Servo On '——伺服 ON。
193 P90CURR = P_Fbc(1) '——获取机器人当前位置。
194  If P90CURR.Z < P1.Z Then '——如果当前位置高度低于原点，则
195 Ovrd 10'——设置速度倍率。
196  P90ESC = P90CURR '——建立一个待机位置
197  P90ESC.Z = P1.Z '——P1 是原点(一般也是待避点)。
198 Mvs P90ESC '——移动待避点。
199  Ovrd 100'——设置速度倍率。
```

```
200 EndIf'——判断语句结束。
201 Mov P1 '——移动到原点
202 Return'——子程序结束。
203  '###追踪中断 ###
     '中断程序的内容是：结束追踪模式，打开抓手，回到待避点。
204  *S91STOP'——程序分支标志。
205  Act 1 = 0'—— 中断程序 1 有效区间终点。
206 Trk Off '——  结束追踪模式。
207  HOpen 1 '——  抓手 OFF。
208 P91P = P_Fbc(1) '——设置 P91P 为当前点。
209 P91P.Z = P1.Z' ——设置 P91P 的 Z 坐标为 P1.Z(原点高度)。
210  Mvs P91P Type 0,0 '——升高机器人。
211  Mov P1 '——回到原点。
212 GoTo *LBFCHK'——跳转回追踪缓存区信息判断。
227 '各位置变量和调节变量的初始值。
    P1：运行前需要设置
    PAC1 = (100.000,100.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PAC11 = (100.000,100.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PAC12 = (100.000,100.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PAC13 = (100.000,100.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PAC2 = (100.000,100.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PAC3 = (100.000,100.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PDLY1 = (1.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PDLY2 = (1.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    POFSET = (0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PPT = (0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PRI = (1.000,1.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PRNG = (300.000,200.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PTN = (1.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PUP1 = (50.000,-50.000,-70.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PUP2 = (0.000,-50.000,-50.000,0.000,0.000,0.000,0.000,0.000)(0,0)
    PWK = (1.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(0,0)
```

29.7 CM1 程序——追踪数据写入程序

本节是对 CM1 程序进行解说。CM1 程序是一个与 1# 程序同时运行的程序。“传送带追踪模式”与“视觉追踪模式”的 CM1 程序是不同的。在不同的模式下选用不同的 CM1 程序。

29.7.1 用于传送带追踪的程序

CM1 程序计算光电开关检测到的工件坐标，该坐标为机器人坐标。这一坐标数据由“A 程序”和“C 程序”获得。CM1 程序将计算完成的数据写入“追踪缓存区”中，“追踪缓存区”是临时存放数据的区域。

1. 获取的数据

(1) 编码器每一脉冲机器人的移动量(P_EncDlt)；

(2) 在“光电开关检测点”与“机器人抓取工件的位置点”这两点的编码器数据之差。

（3）机器人抓取工件的位置。

2. 流程图

CM1 程序是自动程序，放置在任务区 2。

CM1 程序流程图如图 29-39 所示。

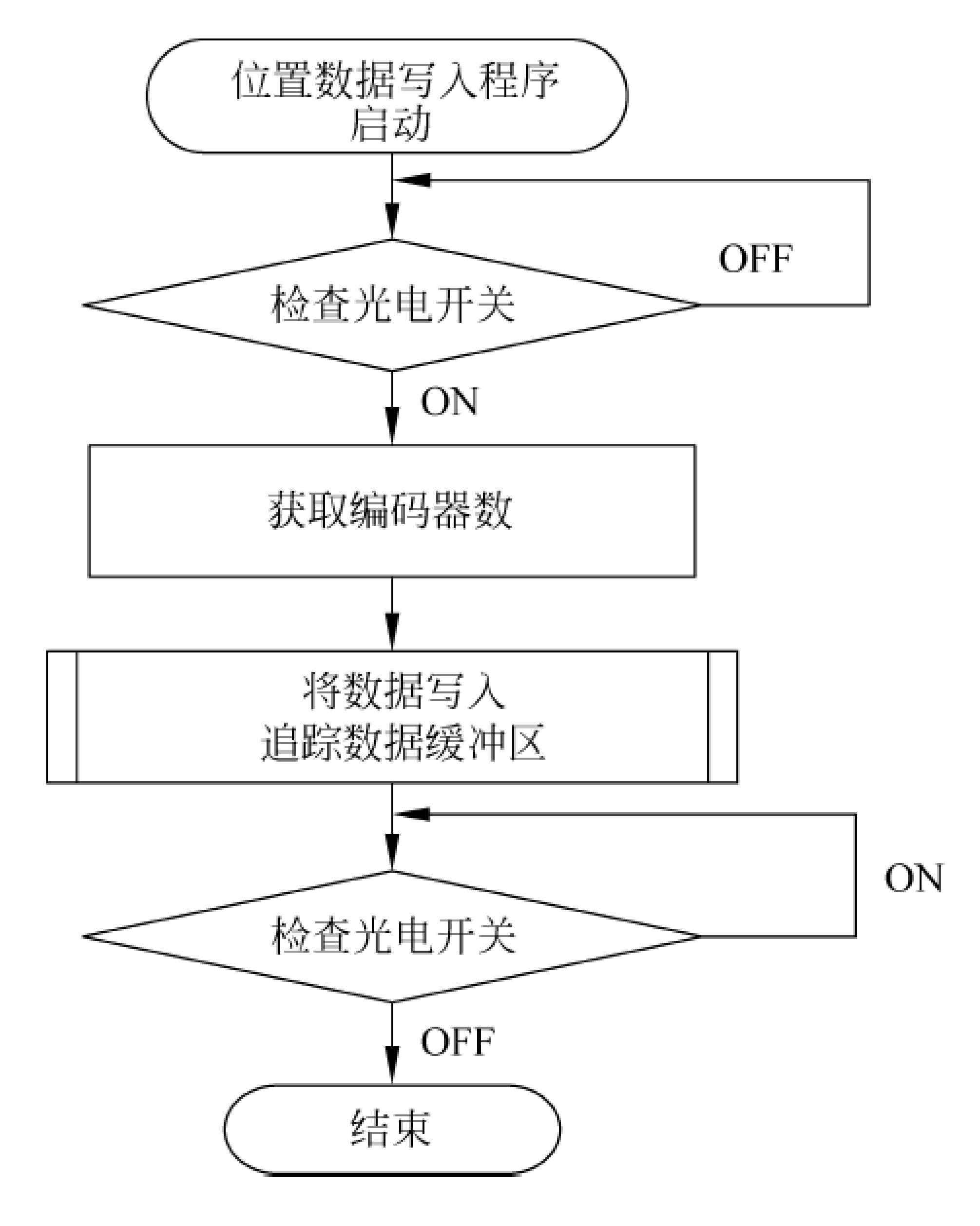


图 29-39　CM1 程序流程图

3. CM1 程序

```
'#####主程序#####
*S00MAIN'——程序分支标志

GoSub *S10DTGET '——获取数据子程序
*LOOP'——程序分支标志
GoSub *S20WRITE '—— 工件位置写入程序
 GoTo *LOOP'——跳转到"*LOOP"行。
 End'——主程序结束。
 '#####获取数据程序#####
 *S10DTGET'——程序分支标志
 '—— 有关抓取工件位置数据、编码器数值、编码器编号已经由 C 程序获得。
  MWKNO=M_09# '——获取工件类型数。
  M10ED# =M_101#(MWKNO) '——编码器数值差
  MENCNO=P_102(MWKNO).X '——编码器编号
  MSNS=P_102(MWKNO).Y '—— 传感器序号
'——计算光电开关检测到的工件位置(X/Y)
PWPOS=P_100(MWKNO)-P_EncDlt(MENCNO)*M10ED# '——
P_100(MWKNO) ——机器人抓取工件位置
P_EncDlt(MENCNO)*M10ED#——从抓取位置到光电检测点的移动量。
PWPOS——光电检测点位置。
```

以上是计算光电检测点位置的计算。

```
 Return'——子程序结束。
'#####位置数据写入程序#####
*S20WRITE'——程序分支标志
 If M_In(MSNS)=0 Then GoTo *S20WRITE '——如果光电开关未动作，则继续等待(M_In(MSNS)是
```

```
光电检测点输入信号),否则(即 M_In(MSNS) = 1,光电开关动作,就执行下一行动作)
MENC# = M_Enc(MENCNO) '——获取此时的编码器数值。
TrWrt PWPOS,MENC#,MWKNO,1,MENCNO '——关键就是这一句:将光电开关检测点的数据写入缓存区。写入的数据有:工件在检测点的位置、检测点的编码器值、工件种类数、缓存区序号、编码器编号。
*L20WAIT'——程序分支标志
If M_In(MSNS) = 1 Then GoTo *L20WAIT'——如果光电开关 = ON,就一直在等待。否则结束子程序。
Return'——子程序结束。
```

29.7.2 用于视觉追踪的 CM1 程序

1. CM1 程序中使用的数据

CM1 程序将视觉系统识别的工件位置写入机器人追踪缓存区。追踪缓存区是临时存放追踪数据的区域。

在 CM1 程序中使用的数据如下。

(1) 编码器每一脉冲机器人的移动量(P_EncDlt)。

(2) 在视觉系统检测点与机器人位置点这两点的编码器数据之差。

(3) 由视觉系统识别的工件位置。

(4) 由视觉系统识别到的工件位置点与机器人抓取点的编码器数据之差。

(5) 工件间距。

(6) 在传送带移动方向上的视觉区域。

(7) 由视觉系统识别的工件长度。

2. 1#程序与 CM1 程序的关系

1#程序追踪传送带上的工件取决于由 CM1 程序写入在追踪缓存区中的工件信息。CM1 程序将被识别的工件位置写入在追踪缓存区中。被存储在追踪缓存区中的信息由 1#程序读出,机器人根据这些信息追踪工件。

1#程序与 CM1 程序的关系可以简述为:

(1) CM1 程序将视觉系统识别的工件位置信息写入“追踪缓存区”。

(2) 1#程序读出追踪缓存区的信息并据此进行追踪工件。

3. 实用 CM1 程序

CM1 程序流程图如图 29-40 所示。

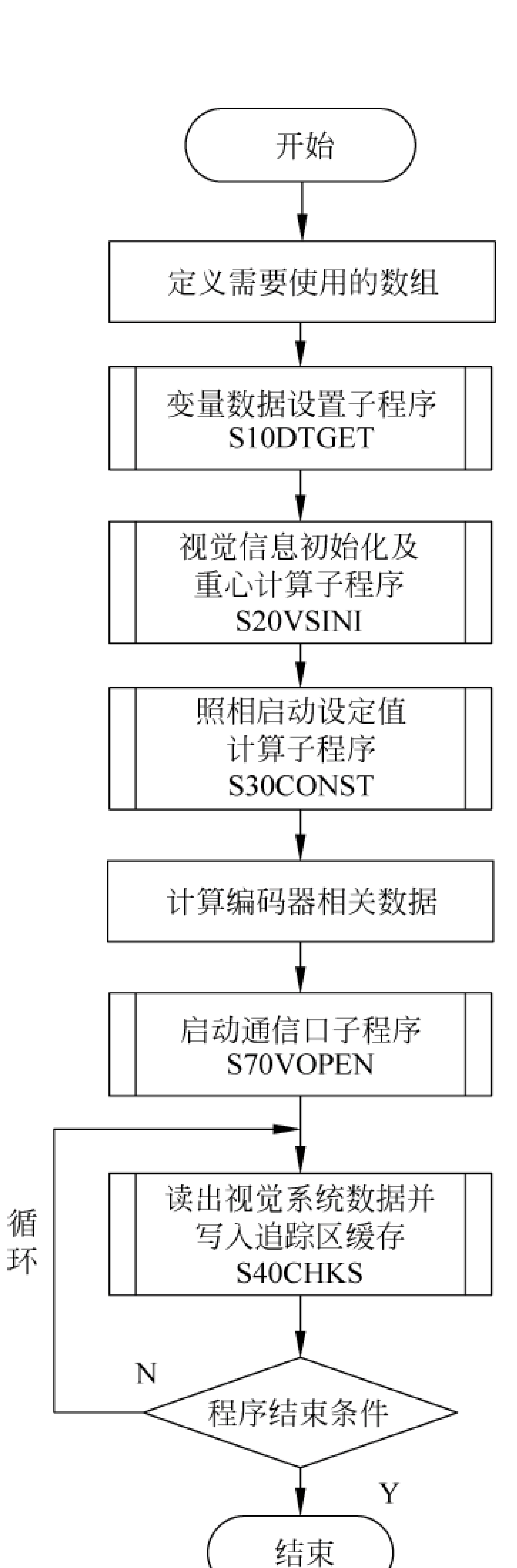


图 29-40 CM1 程序流程图

在以上程序中,最重要的是读数据和写数据的子程序。

程序名为 S40CHKS,其程序流程图如图 29-41 所示。

工件位置与编码器关系如图 29-42 所示。

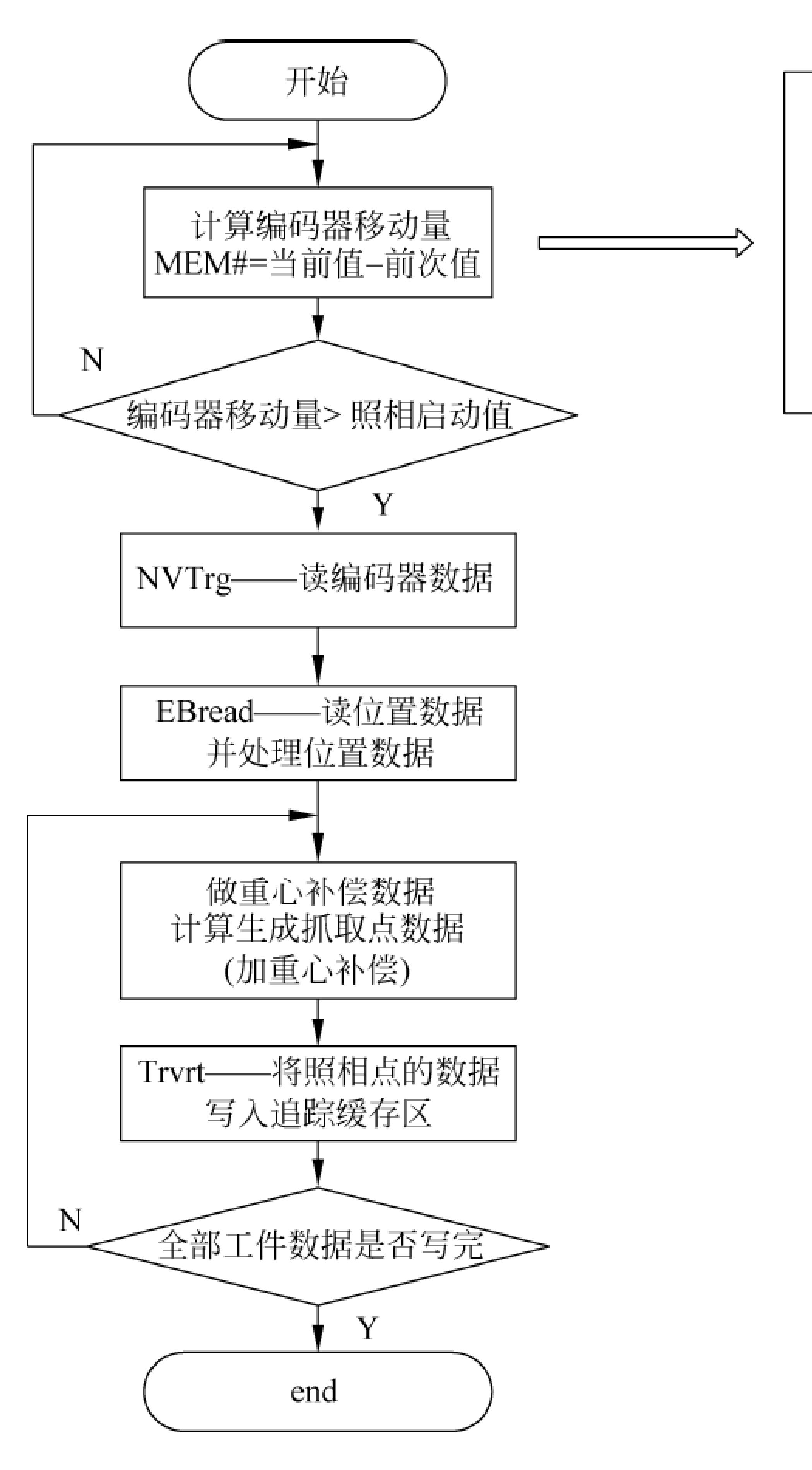


图 29-41　读数据和写数据的流程

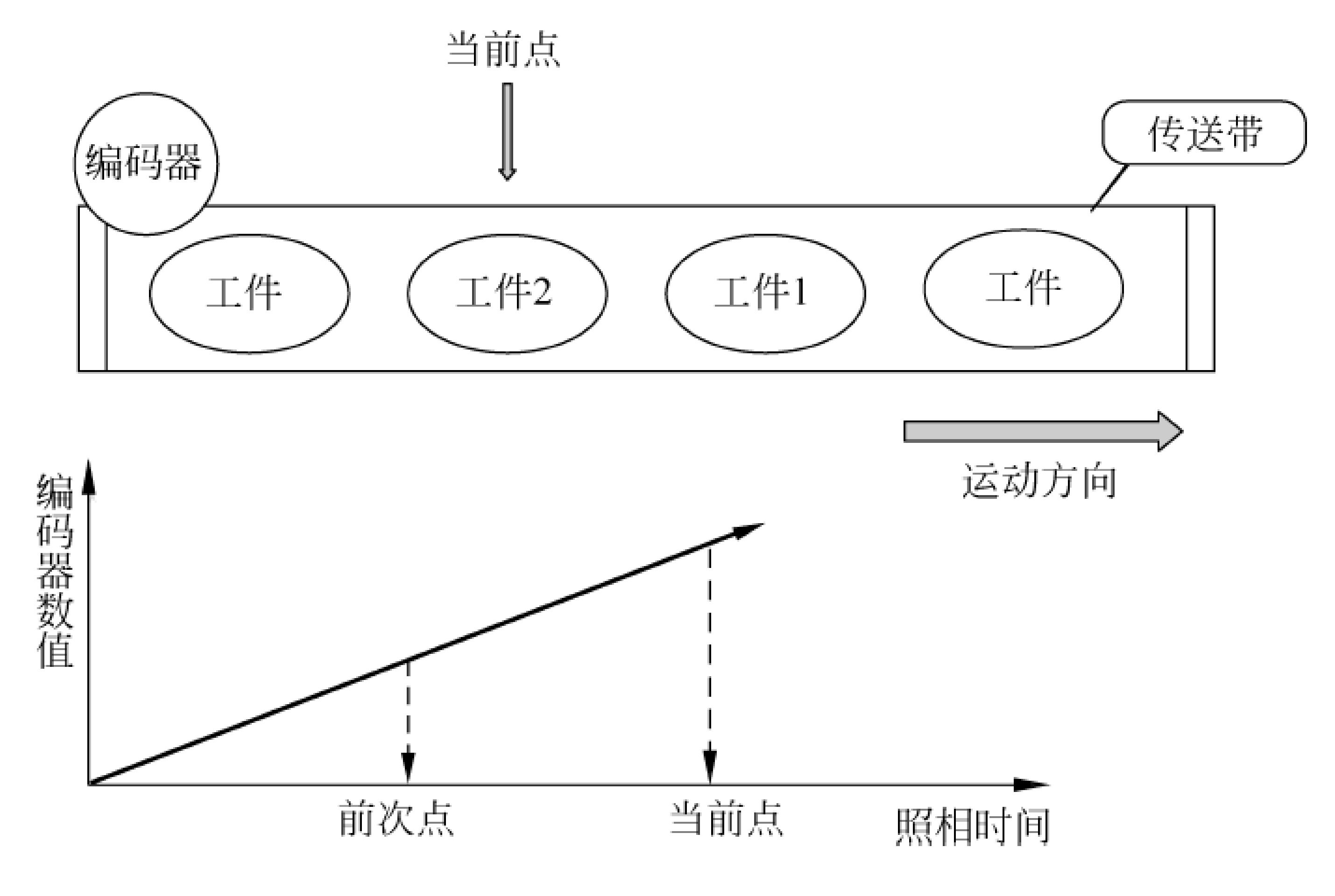


图 29-42　工件位置与编码器数据关系

2）实用 CM1 程序

```
Dim MX(4),MY(4),MT(4),PVS(4)'—— 定义需要使用的数组。
'#####主程序处理#####
*S00MAIN'—— 程序分支标记
 GoSub *S10DTGET  '——调用"数据获取处理"子程序
 GoSub *S20VSINI  '——调用"视觉初始化处理"子程序。
  GoSub *S30CONST  '——调用"条件设定"子程序。
MEP# = M_Enc(MENCNO) + MEI# + 100 '——编码器数据处理。
GoSub *S70VOPEN  '——调用"视觉端口打开+视觉程序加载处理"子程序。
*L00_00'—— 程序分支标记
GoSub *S40CHKS  '——调用"视觉识别检查处理"子程序。
 GoTo *L00_00'——跳转到"*L00_00"行
 End'——主程序结束。
'#####数据处理——对要使用的变量进行赋值#####
 *S10DTGET'—— 程序分支标记
MWKNO = 1  '——M_09#机种编号;
MENCNO = P_102(MWKNO).Y  '——编码器编号;
MVSL = P_103(MWKNO).X  '——VS画面尺寸长边距离;
MWKL = P_103(MWKNO).Y  '——工件尺寸长边距离;
PTEACH = P_100(MWKNO) '——工件抓取点位置;
PVSWRK = P_101(MWKNO)  '——视觉识别的位置;
CCOM$ = C_100$(MWKNO)  '——COM口编号;
CPRG$ = C_101$(MWKNO) '——视觉程序名
Return'——子程序结束
'#####开启通信端口#####
*S70VOPEN'—— 程序分支标记
NVClose  '——端口关闭。
NVOpen CCOM$ As #1  '——端口打开+登录ON。
Wait M_NvOpen(1) = 1  '——等待端口连接完成。
'NVLoad #1,CPRG$  '——视觉程序加载。
Return'—— 子程序结束
'#####视觉系统初始化处理/计算视觉重心与抓取中心偏差,如果超出设定值就报警#####
*S20VSINI'—— 程序分支标记
MED1# = M_100#(MENCNO) '—— M_100# = B程序标定时传送带移动量。
PRBORG = P_EncDlt(MENCNO) * MED1#  '—— PRBORG = 视觉识别点位置
MED2# = M_101#(MWKNO) '—— M_101# = C程序标定时编码器移动量。
PBACK = P_EncDlt(MENCNO) * MED2# '——C程序理论抓取点。
PWKPOS = PRBORG + PVSWRK + PBACK  '——将视觉识别工件变换至机器人区域的位置
PVTR = (P_Zero/PWKPOS) * PTEACH  '——视觉重心位置和夹持位置的向量
If PVTR.X < -PCHK.X Or PVTR.X > PCHK.X  Then Error 9110    '——追踪位置的计算结果与理论值有较大差异时报错。
If PVTR.Y < -PCHK.Y Or PVTR.Y > PCHK.Y  Then Error 9110
Return'—— 子程序结束
'#####条件设定/摄像启动条件计算#####
*S30CONST'—— 程序分支标记
MDX = P_EncDlt(MENCNO).X  '——一个脉冲的移动量(X);
MDY = P_EncDlt(MENCNO).Y  '——一个脉冲的移动量(Y);
MDZ = P_EncDlt(MENCNO).Z  '——一个脉冲的移动量(Z);
MD = Sqr(MDX^2 + MDY^2 + MDZ^2) '——一个脉冲的向量移动量计算。
```

```
MEI# = Abs((MVSL - MWKL)/MD) '——摄像启动设定值计算。
Return'—— 子程序结束
'#####视觉系统识别检查处理#####
*S40CHKS'—— 程序分支标记
*LVSCMD'—— 程序分支标记
*LWAIT'—— 程序分支标记
MEC# = M_Enc(MENCNO) '—— 获取当前编码器值。
MEM# = MEC# - MEP# '——"当前编码器当前值"减去"前次编码器值"。
If MEM#>800000000.0# Then MEM# = MEM# - 1000000000.0#
If MEM#<-800000000.0# Then MEM# = MEM# + 1000000000.0# '——以上是对编码器数值处理。
If Abs(MEM#)>MEI# GoTo *LVSTRG '——编码器移动量和相机启动设定值的比较。
Dly 0.01'——状态 0.01 秒
GoTo *LWAIT'——跳转到"*LWAIT"行
*LVSTRG'—— 程序分支标记
MEP# = MEC#    '—— "编码器当前值"写入"前次值"。
NVTrg #1,5,MTR1#,MTR2#,MTR3#,MTR4#,MTR5#,MTR6#,MTR7#,MTR8#    '——摄像要求+获取编码器值。
'——获取编码器值
If M_NvOpen(1)<>1 Then Error 9100    '——通信异常。
EBRead #1,"",MNUM,PVS1,PVS2,PVS3',PVS4  '——读取视觉系统数据。
pvs11 = PVS1'——赋值计算。
PVS(1) = PVSCal(1,pvs11.X,pvs11.Y,pvs11.C)'——视觉标定。
PVS(1).C = PVS1.C'——赋值计算。
pvs22 = PVS2'——赋值计算。
PVS(2) = PVSCal(1,pvs22.X,pvs22.Y,pvs22.C) '——视觉标定。
PVS(2).C = PVS2.C'——赋值计算。
pvs33 = PVS3'——赋值计算。
PVS(3) = PVSCal(1,pvs33.X,pvs33.Y,pvs33.C) '——视觉标定。
PVS(3).C = PVS3.C'——赋值计算。
pvs44 = pvs4'——赋值计算。
pvs(4) = pvscal(1,pvs44.x,pvs44.y,pvs44.c) '——视觉标定。
If MNUM = 0  Then GoTo *LVSCMD  '——没有检测到工件时跳回子程序起首行。
If MNUM > 4 Then MNUM = 4  '——设工件数为最大 4 个。
For M1 = 1 To MNUM  '——循环次数为识别个数。
MX(M1) = PVS(M1).X   '——数据获取。
MY(M1) = PVS(M1).Y '——数据获取。
MT(M1) = PVS(M1).C '——数据获取。
Next M1 '——下一循环。
GoSub *S60WRDAT  '——调用"识别数据存储处理"子程序。
Return'——子程序结束。
'#####追踪数据存储处理#####
*S60WRDAT'—— 程序分支标记
For M1 = 1 To MNUM '——处理次数为识别个数。
PSW = P_Zero'——初始化置零。
PSW = PRBORG  '—— PRBORG 为"B 程序确定的理论视觉点"。
PSW.X = PSW.X + MX(M1) '——生成视觉识别位置。
PSW.Y = PSW.Y + MY(M1) '——生成视觉识别位置。
PSW.C = PSW.C - MT(M1) '——生成视觉识别位置。
PRW = P_Zero'——初始化置零。
PVTR.X =- 40'——赋值。
```

```
PVTR.Y=-72'——赋值。
PFIX=P_Zero'——初始化置零。
PFIX.X=8'——赋值。
PFIX.Y=-13'——赋值。
PVVV=PVTR*PFIX'——计算(两点乘法运算)
  PRW=PSW+PVVV  '——最终获得的视觉点 PRW 计算公式。
    PRW=B 程序理论点+视觉补偿+重心补偿
  PRW.FL1=P_100(MWKNO).FL1 '——赋值
  PRW.FL2=P_100(MWKNO).FL2'——赋值
  Select MENCNO '——根据编码器编号选择执行不同的写入指令。
  Case 1'——MENCNO=1
    TrWrt PRW,MTR1#,MWKNO,1,MENCNO  '——(写入"识别点"及"编码器数值")变量依次为:位置,编码器值,工件种类编号,缓存编号,编码器编号。
  Break'——结束选择语句"Select MENCNO"。
  Case 2'——MENCNO=2
  TrWrt PRW,MTR2#,MWKNO,1,MENCNO   '——位置,编码器值,机种编号,缓存编号,编码器编号
    Break'——结束选择语句"Select MENCNO"。
 Case 3'——MENCNO=3
    TrWrt PRW,MTR3#,MWKNO,1,MENCNO '——位置,编码器值,机种编号,缓存编号,编码器编号。
    Break'——结束选择语句"Select MENCNO"。
 Case 4'——MENCNO=4
  TrWrt PRW,MTR4#,MWKNO,1,MENCNO  '——位置,编码器值,机种编号,缓存编号,编码器编号
  Break'——结束选择语句"Select MENCNO"。
 Case 5'——MENCNO=5
    TrWrt PRW,MTR5#,MWKNO,1,MENCNO  '——位置,编码器值,机种编号,缓存编号,编码器编号
     Break'——结束选择语句"Select MENCNO"。
 Case 6'——MENCNO=6
    TrWrt PRW,MTR6#,MWKNO,1,MENCNO  '——位置,编码器值,机种编号,缓存编号,编码器编号。
  Break'——结束选择语句"Select MENCNO"。
    Case 7'——MENCNO=7
  TrWrt PRW,MTR7#,MWKNO,1,MENCNO  '——位置,编码器值,机种编号,缓存编号,编码器编号。
    Break'——结束选择语句"Select MENCNO"。
 Case 8'——MENCNO=8
    TrWrt PRW,MTR8#,MWKNO,1,MENCNO '——位置,编码器值,机种编号,缓存编号,编码器编号。
    Break'——结束选择语句"Select MENCNO"。
  End Select'——结束选择语句"Select MENCNO"。
  Next M1'——进入下一循环。
 Return'—— 子程序结束
PVS(1)=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PVS(2)=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PVS(3)=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PVS(4)=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PTEACH=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PCSWRK=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PRBORG=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PBACK=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PWKPOS=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PVTR=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PCHK=(+100.000,+100.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PSW=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
PRW=(+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)(0,0)
```

```
PVS(1) = ( + 1012.661, + 648.378, + 0.000, + 0.000, + 0.000, + 179.511, + 0.000, + 0.000)(0,0)
PVS(2) = ( + 1012.661, + 648.378, + 0.000, + 0.000, + 0.000, + 179.511, + 0.000, + 0.000)(0,0)
PVS(3) = ( + 1012.661, + 648.378, + 0.000, + 0.000, + 0.000, + 179.511, + 0.000, + 0.000)(0,0)
PVS(4) = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(,)
PTEACH = ( + 670.618, - 41.811, + 217.342, + 0.000, + 0.000, - 74.196, + 0.000, + 0.000)(0,0)
PVSWRK = ( + 550.148, + 20.165, + 0.000, + 0.000, + 0.000, + 179.511, + 0.000, + 0.000)(0,0)
PRBORG = ( + 1224.914, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(4,0)
PBACK = ( - 1070.989, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(4,0)
PWKPOS = ( + 704.073, + 20.165, + 0.000, + 0.000, + 0.000, + 179.511, + 0.000, + 0.000)(4,0)
PVTR = ( + 32.925, + 62.260, + 217.342, + 0.000, + 0.000, + 106.293, + 0.000, + 0.000)(0,0)
PCHK = ( + 900.000, + 800.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(0,0)
PVS1 = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(0,0)
PVS2 = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(0,0)
PVS3 = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(0,0)
pvs11 = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(0,0)
pvs22 = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(0,0)
pvs33 = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(0,0)
PSW = ( + 1695.076, - 170.838, + 0.000, + 0.000, + 0.000, - 10332.774, + 0.000, + 0.000)(4,0)
PRW = ( + 1654.076, - 243.838, + 217.342, + 0.000, + 0.000, - 10226.480, + 0.000, + 0.000)(0,0)
PFIX = (0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(,)
PVVV = (0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000)(,)
PCSWRK = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(,)
pvs0 = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(0,0)
PVS4 = ( + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000, + 0.000)(,)
```

29.8 需要思考的问题

(1) A 程序有什么功能？运行 A 程序后得到什么结果？

(2) B 程序有什么功能？运行 B 程序后得到什么结果？

(3) C 程序有什么功能？运行 C 程序后得到什么结果？

(4) CM 程序有什么功能？CM 程序在什么情况下运行？

(5) 1＃程序有什么功能？

第 30 章 第 30 日——机器人编程软件 RT ToolBox 的学习使用

【学习目的】

RT ToolBox2 软件(以下简称 RT)是一款专门用于三菱机器人编程、参数设置、程序调试、工作状态监视的软件。其功能强大,编程方便。在前面的学习中,已经介绍过使用 RT 软件设置参数,编制简短程序的方法,本章对 RT 软件的使用做一系统的介绍(RT ToolBox2 软件可以在 http://cn.mitsubishielectric.com/fa/zh/网站下载)。

30.1 RT 软件的基本功能

30.1.1 RT 软件的功能概述

1. RT 软件的 5 大功能

(1) 编程及程序调试功能;

(2) 参数设置功能;

(3) 备份还原功能;

(4) 工作状态监视功能;

(5) 维护功能。

2. RT 软件的三种工作模式

(1) 离线模式;

(2) 在线模式;

(3) 模拟模式。

30.1.2 RT 软件的功能一览

RT 软件的功能如表 30-1 所示。

表 30-1 RT 软件的基本功能

功　　能	说　　明
离线——以计算机中的工程作为对象(不连接机器人控制器)	
机器人机型名称	显示要使用的机器人机型名称
程序	编制程序
样条	编制样条曲线
参数	设置参数。在与机器人连接后传入机器人控制器

续表

功　能	说　明
在线——以机器人控制器中的工程作为对象(连接机器人控制器)	
程序	编制程序
样条	编制样条曲线
参数	设置参数
在线——监视(监视机器人工作状态)	
动作监视	可以监视任务区状态、运行的程序、动作状态、当前发生报警
信号监视	监视机器人的输入输出信号状态
运行监视	监视机器人运行时间、各个机器人程序的生产信息
在线——维护	
原点数据	设定机器人的原点数据
初始化	进行时间设定、程序全部删除、电池剩余时间的初始化、机器人的序列号的设定
位置恢复支持	进行原点位置偏差的恢复
TOOL 长自动计算	自动计算 TOOL 长度,设定 TOOL 参数
伺服监视	进行伺服电机工作状态的监视
密码设定	密码的登录/变更/删除
文件管理	能够对机器人遥控器内的文件进行复制、删除、变更名称
2D Vision Calibration	2D 视觉标定
在线——选项卡	
在线——TOOL	
力觉控制	
用户定义画面编辑	
示波器	
模拟	
模拟	完全模拟在线状态
节拍时间测定	
备份-还原	
备份	从机器人控制器传送工程文件到计算机
还原	从计算机传送工程文件到机器人控制器
MELFA 3D-Vision	能够进行 MELFA-3D Vision 的设定和调整

30.2　程序的编制调试管理

30.2.1　编制程序

由于使用本软件有“离线”和“在线”模式,大多数编程是在离线模式下完成的,在需要调试和验证程序时则使用在线模式。在离线模式下编制完成的程序要首先保存在计算机里,在调试阶段,连接到机器人控制器后再选择在线模式,将编制完成的程序写入“机器人控制器”。所以以下叙述的程序编制等全部为离线模式。

1. 工作区的建立

工作区就是一个总项目。

工程就是总项目中每一台机器人的工作内容(程序、参数)。一个工作区内可以设置32个工程,也就是管理32台机器人。新建一个工作区的方法如下。

(1) 打开RT软件。

(2) 单击"工作区"→"新建",弹出如图30-1所示的"工作区的新建"窗口,如图30-1所示,设置"工作区名"和"标题",单击OK按钮。这样,一个新工作区设置完成。同时弹出如图30-2所示的"工程编辑"对话框。



图30-1 "工作区的新建"窗口

2. 工程的新建

工程就是总项目中每一台机器人的工作内容(程序、参数),所以需要设置的内容如图30-2所示。

(1) 工程名称;

(2) 机器人控制器型号;

(3) 与计算机的通信方式(如USB、以太网);

(4) 机器人型号;

(5) 机器人语言;

(6) 行走台工作参数设置。

在一个工作区内可以设置32个工程。如图30-3所示,在一个工作区内设置了4个工程。

3. 程序的编辑

编辑程序时,菜单栏中会追加"文件""编辑""调试""工具"项目。各项目所含的内容如下。

1) "文件"菜单

"文件"菜单所含项目如表30-2所示。

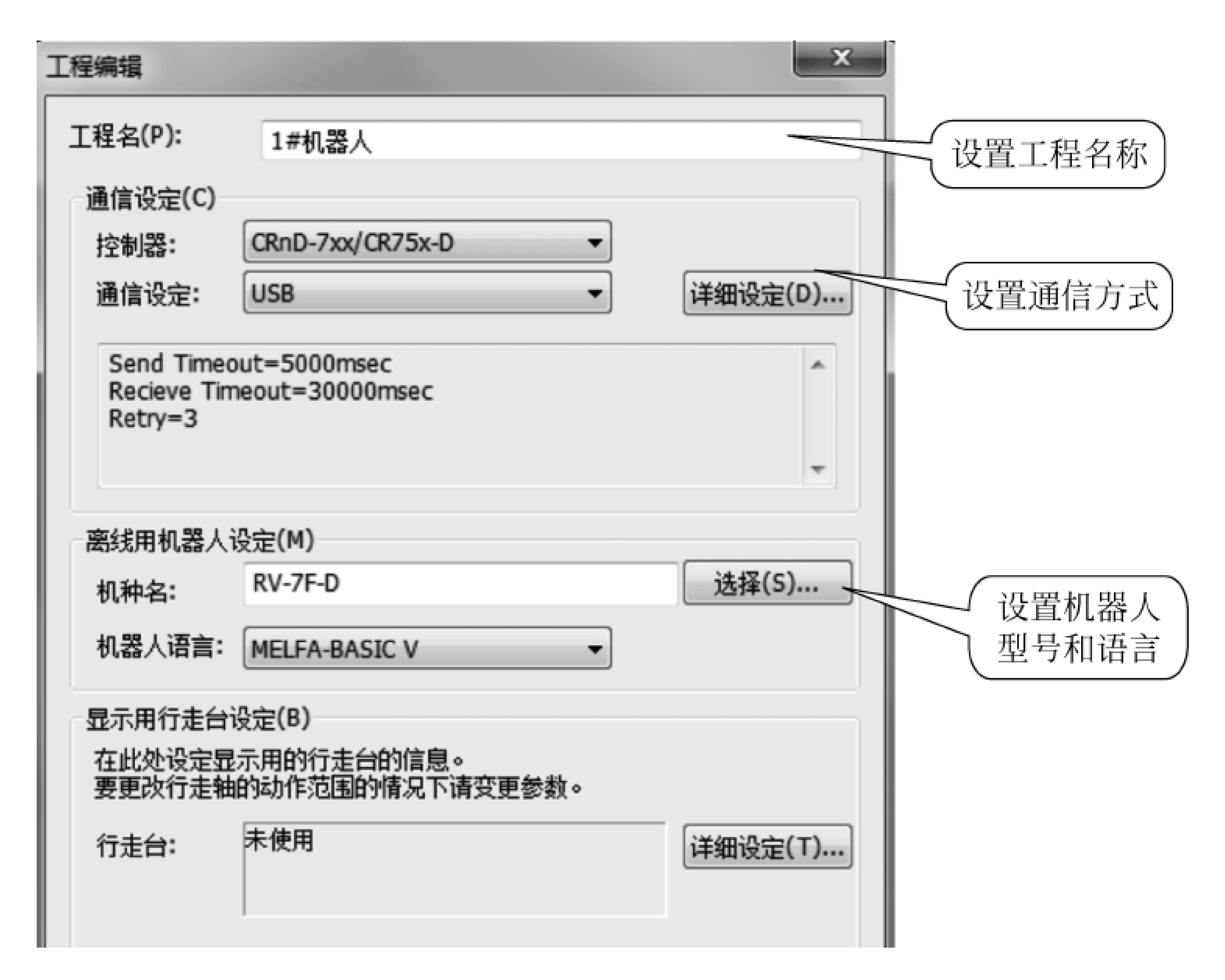


图 30-2 “工程编辑”对话框

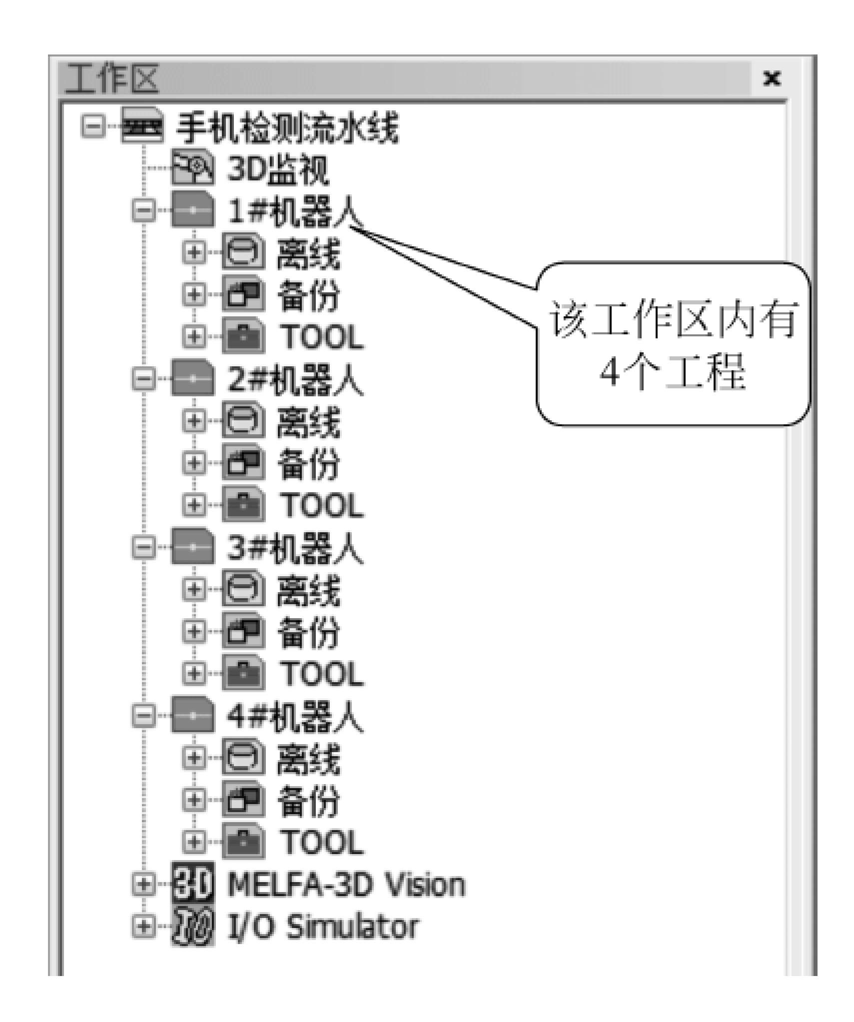


图 30-3 一个工作区内设置了 4 个工程

表 30-2 “文件”菜单

菜单项目(文件)	项　　目	说　　明
覆盖保存(S)　Ctrl+S	覆盖保存	以现有程序覆盖原程序
保存在电脑上(A)...	保存在电脑上	将编辑中的程序保存在计算机
保存到机器人上(T)...	保存到机器人上	将编辑中的程序保存到机器人控制器
页面设定(U)...	页面设定	设置打印参数

2）“编辑”菜单

“编辑”菜单所含项目如表 30-3 所示。

表 30-3 “编辑”菜单

菜单项目（编辑）	项　　目	说　　明
	还原	撤销本操作
	Redo	恢复原操作（前进一步）
	还原-位置数据	撤销本位置数据
	Redo-位置数据	恢复-位置数据（前进一步）
	剪切	剪切选中的内容
	复制	复制选中的内容
	粘贴	把复制、剪切的内容粘贴到指定位置
	复制-位置数据	对位置数据进行复制
	粘贴-位置数据	对复制的位置数据进行状态
	检索	查找指定的字符串
	从文件检索	在指定的文件中进行查找
	替换	执行替换操作
	跳转到指定行	跳转到指定的程序行号
	全写入	将编辑的程序全部写入机器人控制器
	部分写入	将编辑程序的选定部分写入机器人控制器
	选择行的注释	将选择的程序行变为“注释行”
	选择行的注释解除	将“注释行”转为程序指令行
	注释内容的统一删除	删除全部注释
	命令行编辑-在线	调试状态下编辑指令
	命令行插入-在线	调试状态下插入指令
	命令行删除-在线	调试状态下删除指令

3）“调试”菜单

“调试”菜单所含项目如表 30-4 所示。

表 30-4 “调试”菜单

菜单项目（编辑）	项　　目	说　　明
	设定断点	设定单步执行时的“停止行”
	解除断点	解除对“断点”的设置
	解除全部断点	解除对全部“断点”的设置
	总是显示执行行	在执行行显示光标

4）“工具”菜单

“工具”菜单所含项目如表 30-5 所示。

表 30-5 “工具”菜单

菜单项目(编辑)	项　　目	说　　明
工具(T) 窗口(W) 帮助(H) 重新编号(R)... 排列(S) 语法检查(Y) 指令模板(C)... 直交位置数据统一编辑(X)... 关节位置数据统一编辑(J)... 节拍时间测量(T)... 选项(O)...	语法检查	对编辑的程序进行“语法检查”
	指令模板	提供标准指令格式供编程使用
	直交位置数据统一编辑	对“直交位置数据”进行统一编辑
	关节位置数据统一编辑	对“关节位置数据”进行统一编辑
	节拍时间测量	在模拟状态下对选择的程序进行运行时间测量
	选项	设置编辑的其他功能

4. 新建和打开程序

1）新建程序

在“工程树”单击“程序”→“新建”，弹出程序名设置框。设置程序名后，弹出编程框如图 30-4 所示。

2）打开

在“工程树”中单击“程序”，弹出原有排列程序框。选择程序名后，单击“打开”，弹出编程框如图 30-4 所示。

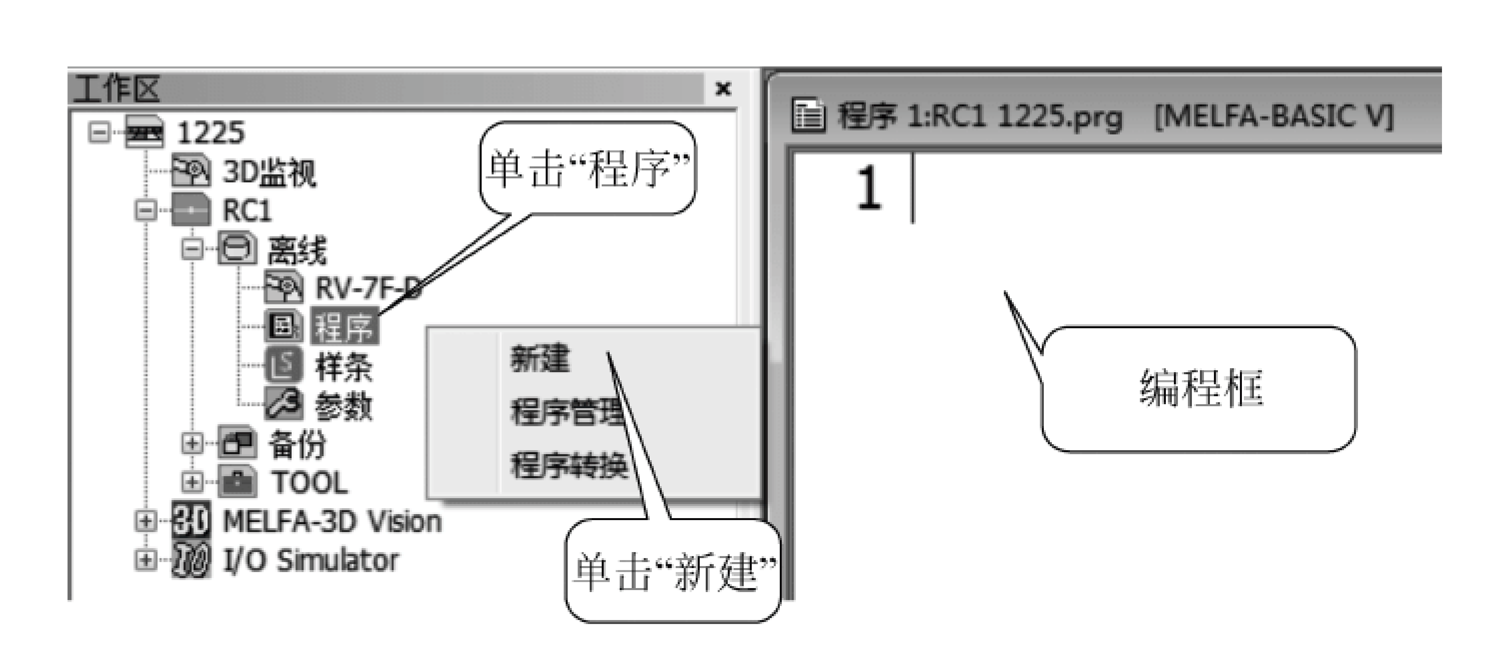


图 30-4 新建及打开编程框

5. 编程注意事项

(1) 无须输入程序行号，软件自动生成程序行号。

(2) 输入指令不区分大小写字母，软件自动转换。

(3) 直交位置变量、关节位置变量在各自编辑框内编辑；位置变量的名称不区分大小写字母。位置变量在编辑时，有“追加”“变更”“删除”等按键。

(4) 编辑中的辅助功能，如剪切、复制、粘贴、检索(查找)、替换，与一般软件的使用方法相同。

(5) 位置变量的统一编辑——本功能用于对于大量的位置变量需要统一修改某些轴的变量(可以加减或直接修改)的场合，可用于机械位置发生相对移动的场合。单击“工具”→“位置变量统一变更”就弹出如图 30-5 所示的对话框。

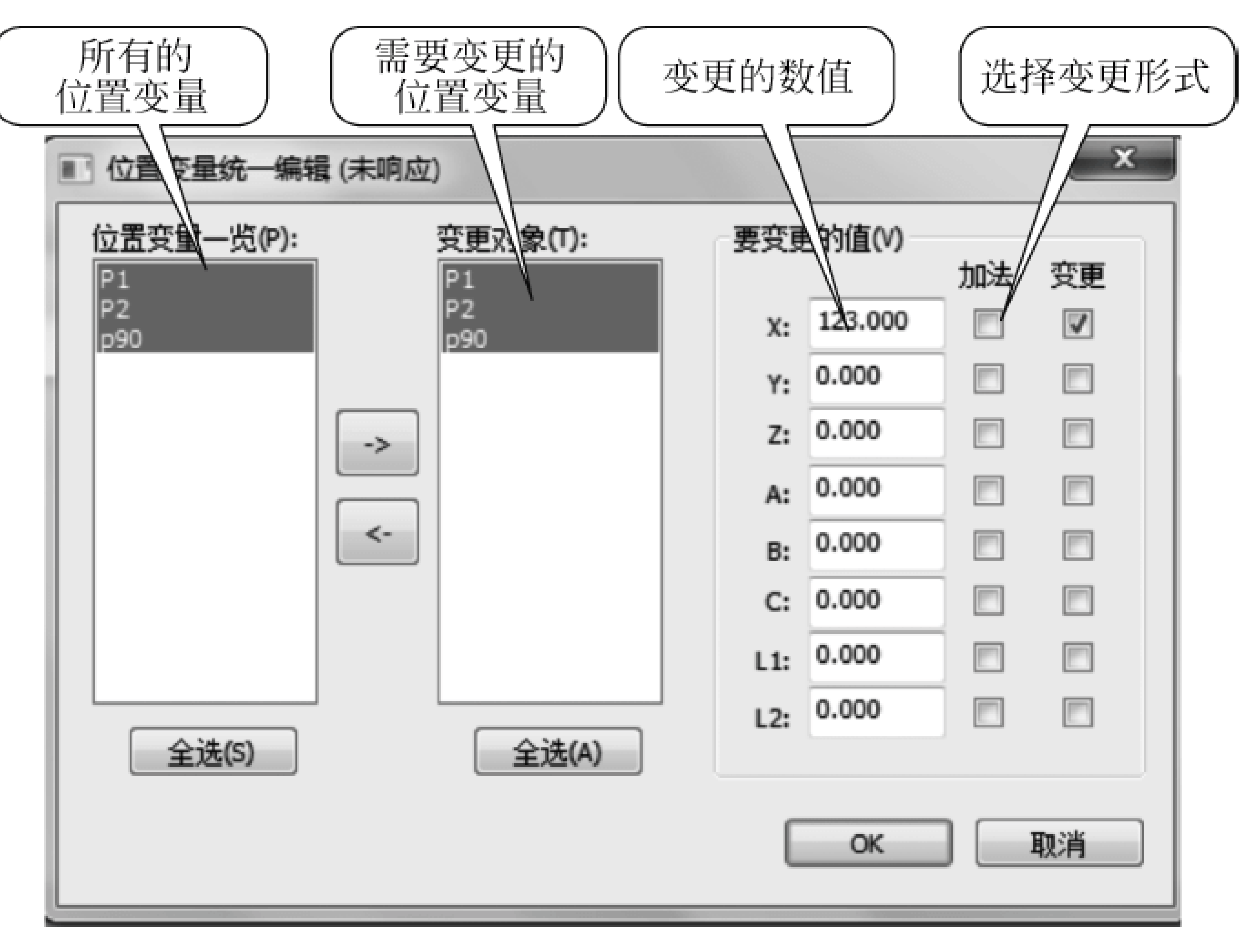


图 30-5 "位置变量统一编辑"对话框

(6) 全写入。

本功能是将"当前程序"写入机器人控制器中。单击菜单"编辑"→"全写入"。在确认信息显示后,单击"是"。这是本软件特有的功能。

(7) 语法检查。

语法检查用于检查所编辑的程序在语法上是否正确。在向控制器写入程序前执行。单击菜单栏中的"工具"→"语法检查"。在语法上有错误的情况下,会显示发生错误的程序行和错误内容,如图 30-6 所示。语法检查功能是经常使用的。



图 30-6 语法检查报警框

(8) 指令模板。

指令模板就是"标准的指令格式"。如果编程者记不清楚程序指令,可以使用本功能。本功能可以显示全部的指令格式,只要选中该指令双击后就可以插入到程序指令编辑位置。

使用方法:单击菜单栏中的"工具"→"指令模板",弹出如图 30-7 所示的"指令模板"对话框。

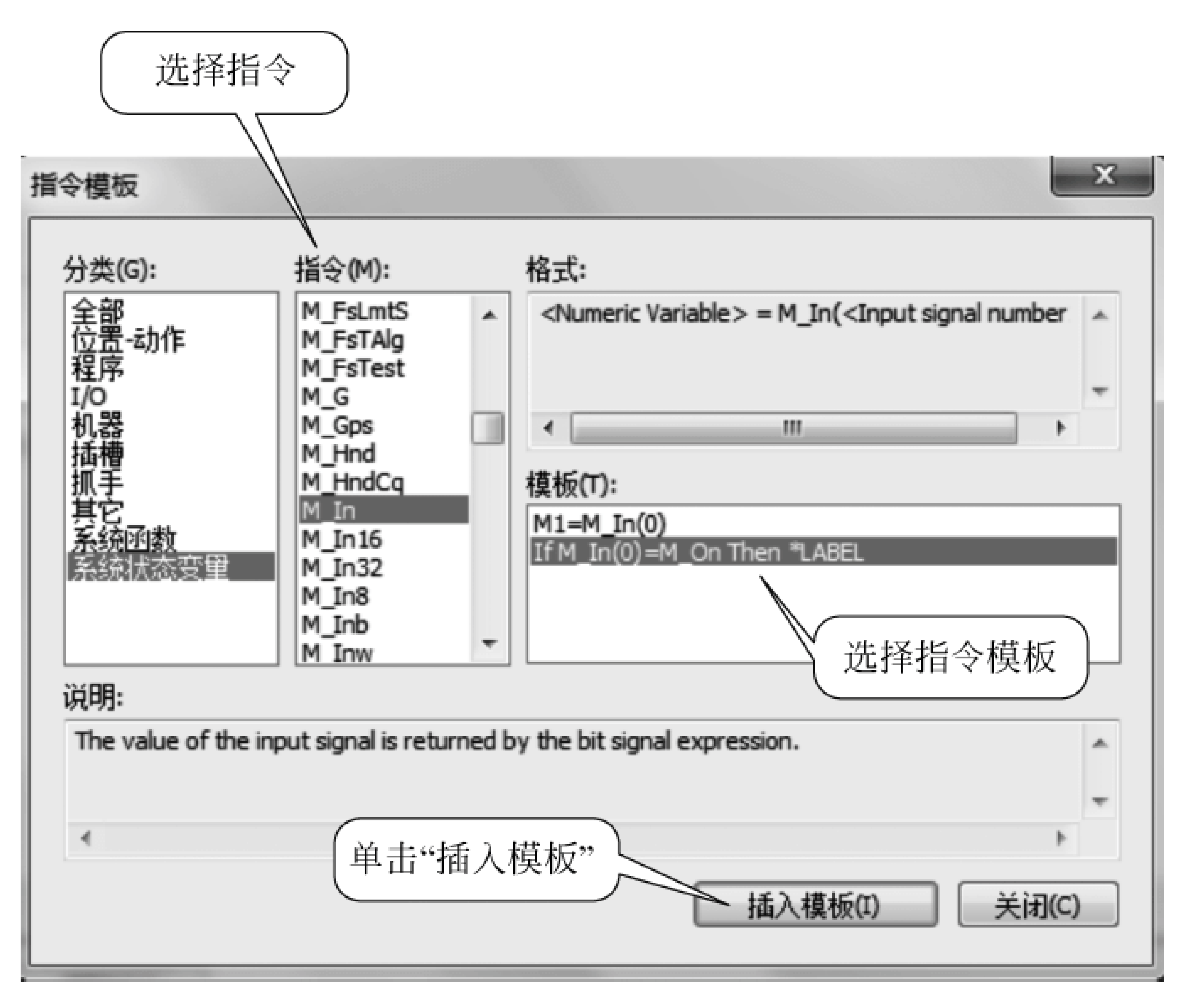


图 30-7 “指令模板”对话框

(9) 选择行的注释/选择行的注释解除。

本功能是将某一程序行变为“注释文字”或解除这一操作。在实际编程中，特别是对于使用中文进行程序注释时，可能会一行一行先写中文注释，最后再写程序指令。因此，可以先写中文注释，然后使用本功能将其全部转为“注释信息”。这是简便的方法之一。

在指令编辑区域中，选中要转为注释的程序行，单击菜单栏中的“编辑”→“选择行的注释”。选中的行的开头会加上注释文字标志“’”，变为注释信息。另外，选中需要解除注释的行后，再单击菜单栏中的“编辑”→“选择行的注释解除”，就可以解除选择行的注释。

6. 位置变量的分类

位置变量的编辑是最重要的工作之一。位置变量分为直交型变量和关节型变量。在进行位置变量编辑时，首先要分清是“直交型变量”还是“关节型变量”，如图 30-8 所示。

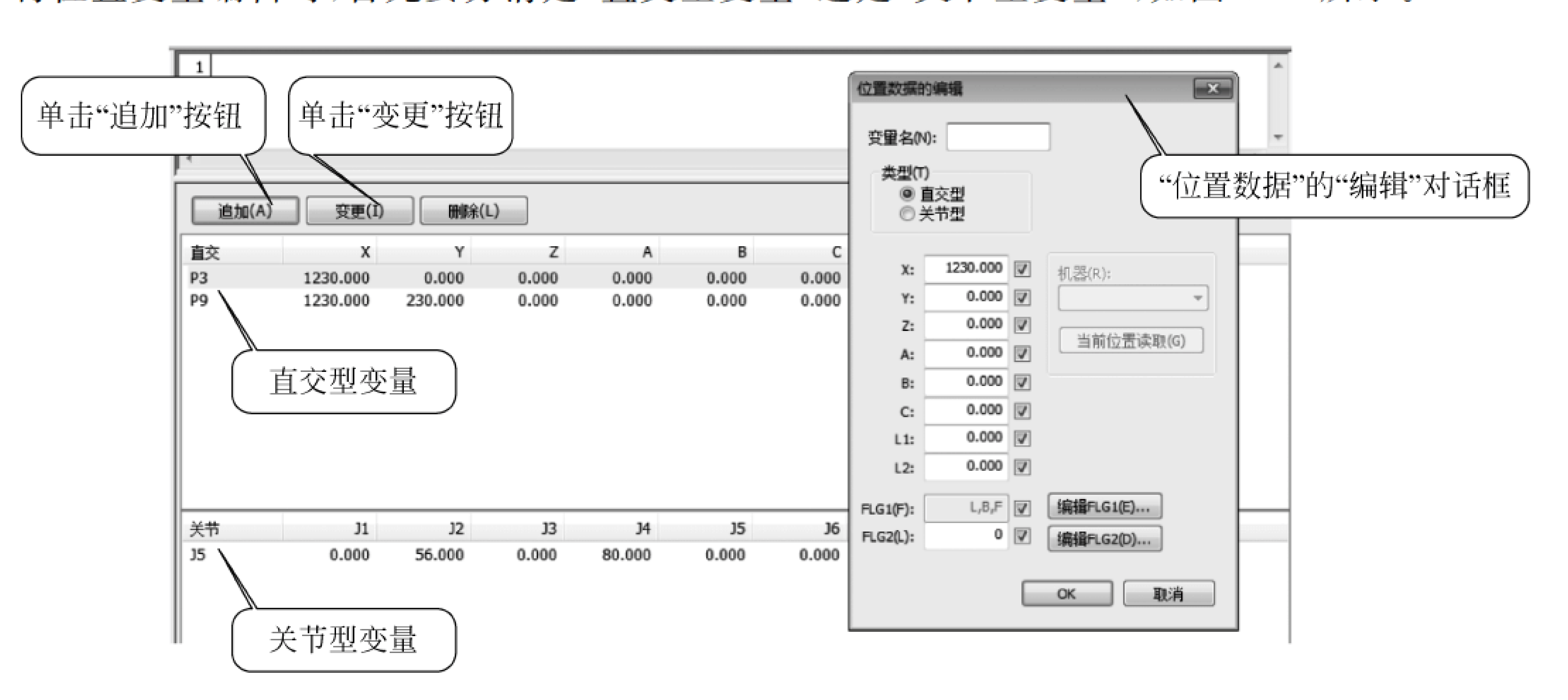


图 30-8 位置变量编辑

7. 位置变量的编辑

首先区分是位置变量还是关节变量，如果要增加一个新的位置点，单击“追加”按钮，弹出“位置数据的编辑”对话框，如图30-8所示。在“位置数据的编辑”对话框中需要设置以下项目。

1）设置变量名称

（1）直交型变量设置为P***，注意以P开头，如P1，P2，P10。

（2）关节型变量设置为J***，注意以J开头，如J1，J2，J10。

2）选择变量类型

选择是直交型变量还是关节型变量。

3）设置位置变量的数据

设置位置变量的数据有以下两种方法。

（1）读取当前位置数据——当使用示教单元移动到“工作目标点”后，直接单击“当前位置读取”按钮，在左侧的数据框中立即自动显示“工作目标点”的数据，单击OK按钮，即设置了当前的位置点。这是常用的方法之一。

（2）直接设置数据——根据计算，直接将数据设置到对应的数据框中。单击OK按钮，即设置了位置点数据。如果能够用计算方法计算运行轨迹，则用这种方法。

4）数据修改

如果需要修改“位置数据”，操作方法如下。

（1）选定需要修改的数据。

（2）单击“变更”按钮，弹出如图30-9所示“位置数据的编辑”对话框。

（3）修改位置数据。

（4）单击OK按钮，数据修改完成。

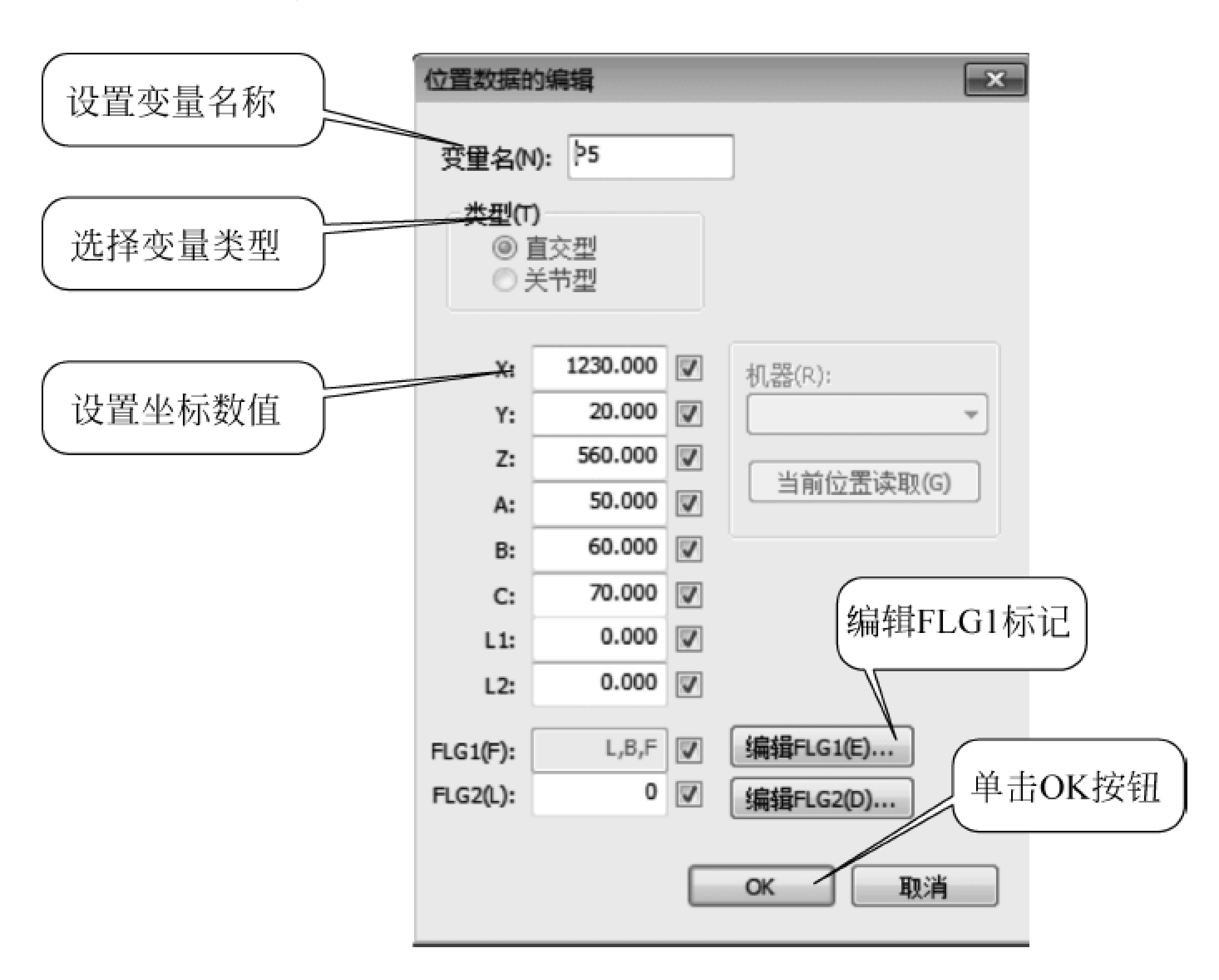


图30-9　直接设置数据

5）数据删除

如果需要删除位置数据，操作方法如下，如图 30-8 所示。

（1）选定需要删除的数据。

（2）单击“删除”按钮，单击 YES。

（3）数据删除完成。

8. 编辑辅助功能

单击“工具”→“选项”，弹出“选项”对话框，如图 30-10 所示。该对话框有以下功能。

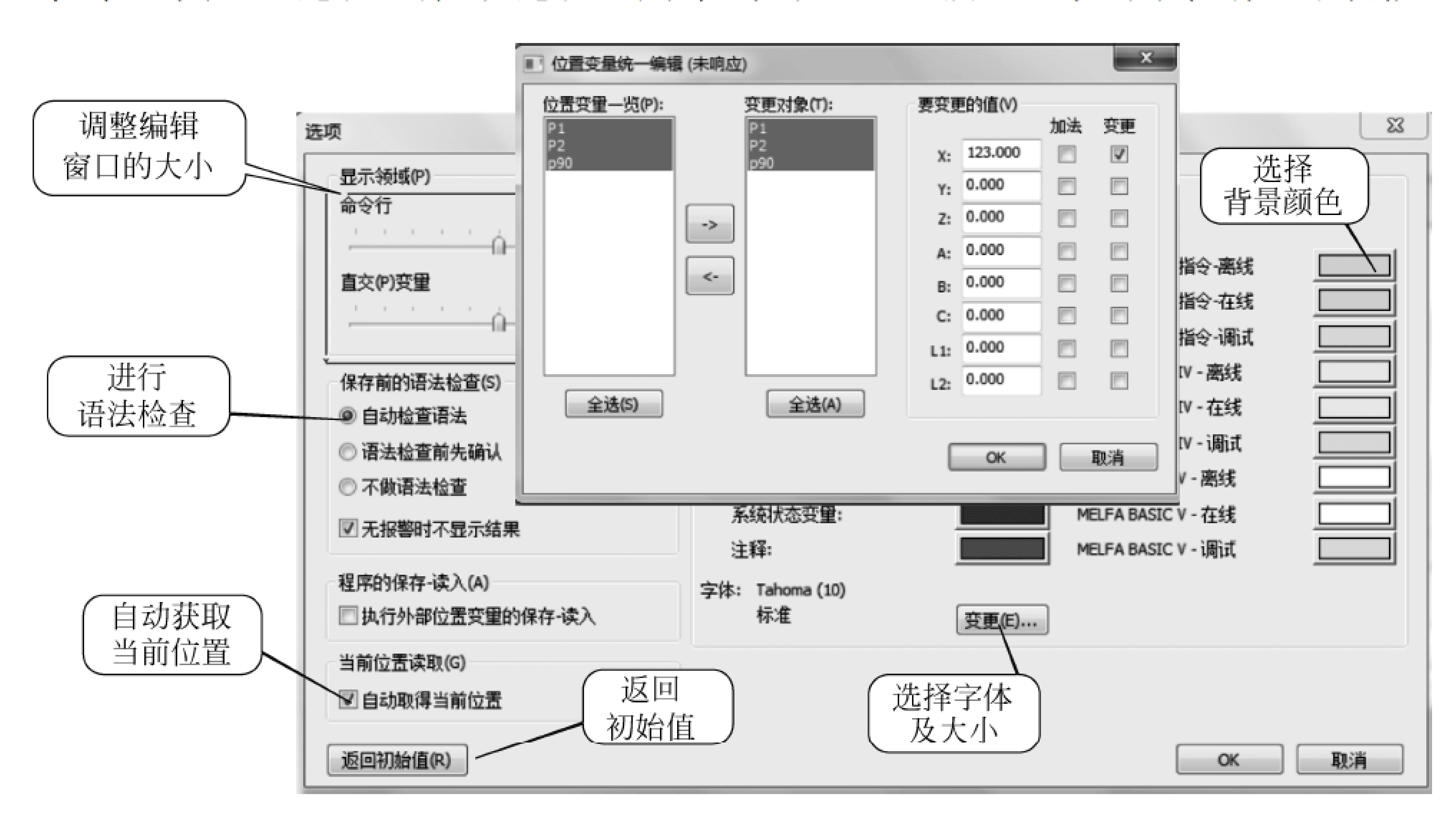


图 30-10 “选项”对话框

（1）调节各分区的大小，即调节程序编辑框、直交位置数据编辑框、关节位置数据编辑框的大小。

（2）对编辑指令的语法检查进行设置；对编辑指令的正确与否进行自动检查，可在写入机器人控制器之前，自动进行语法检查并提示。

（3）对“自动取得当前位置”进行设置。

（4）返回初始值的设置；如果设置混乱后，可以回到初始值重新设置。

（5）对指令颜色的设置。为视觉方便，对不同的指令类型、系统函数、系统状态变量标以不同的颜色。

（6）对字体类型及大小的设置。

（7）对背景颜色的设置。为视觉方便可以对屏幕设置不同的背景颜色。

9. 程序的保存

1）覆盖保存

用当前程序“覆盖”原来的（同名）程序并保存。

单击菜单栏中的“文件”→“覆盖保存”后，进行覆盖保存。

2）保存到计算机

将当前程序保存到计算机上。应该将程序经常保存到计算机上，以免丢失。单击菜单栏中的“文件”→“保存在电脑上”。

3）保存到机器人

在计算机与机器人连线后，将当前编辑的程序保存到机器人控制器。调试完毕一个要执行的程序后当然是要保存到机器人控制器。

单击菜单栏中的“文件”→“保存在机器人上”。

30.2.2 程序的管理

1. 程序管理

程序管理是指以“程序”为对象，对“程序”进行复制、移动、删除、重新命名、比较等操作。操作方法如下。

单击“程序”→“程序管理”，弹出如图30-11所示的“程序管理”窗口。

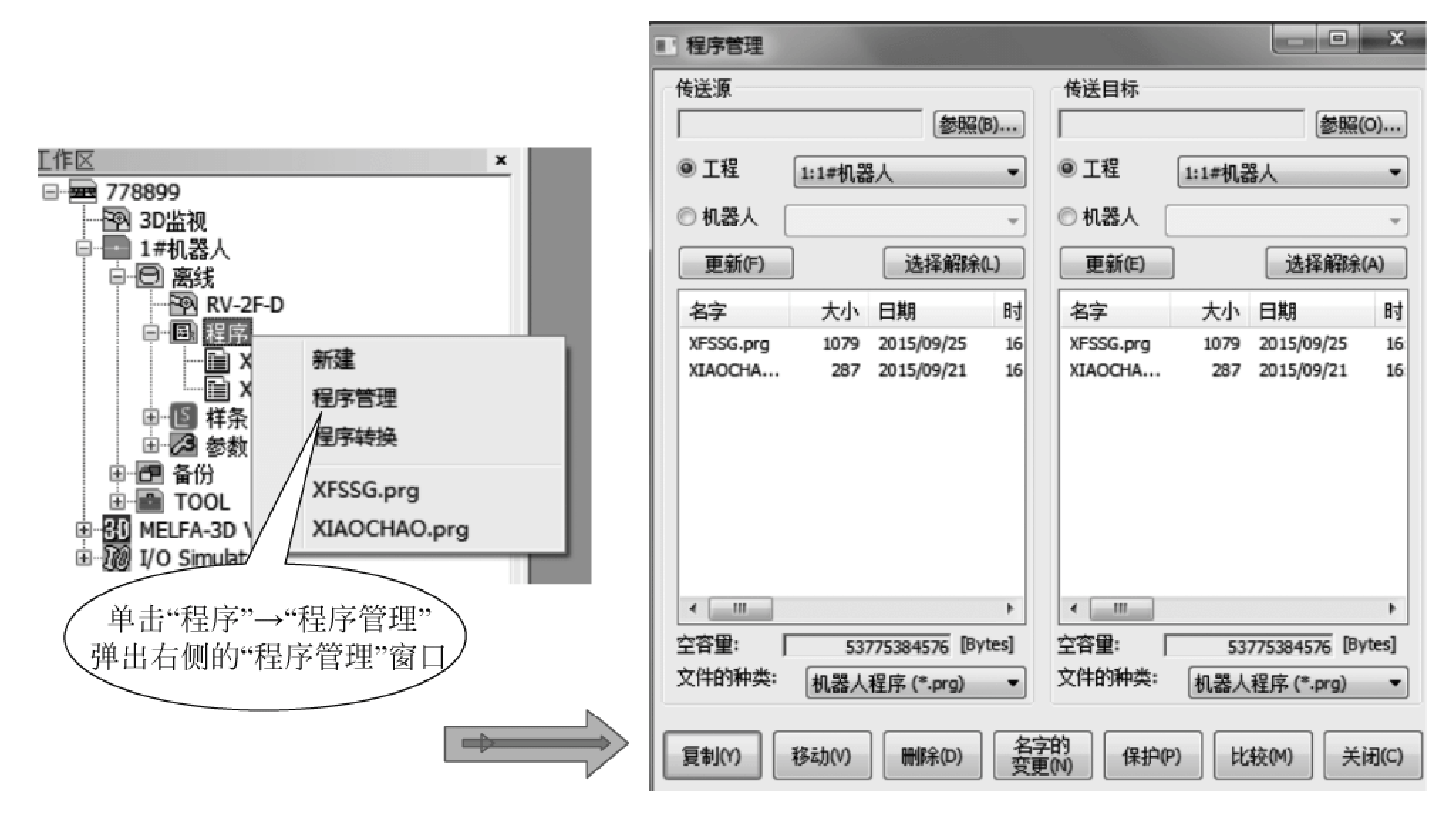


图30-11 “程序管理”窗口

“程序管理”窗口分为左右两部分，如图30-12所示，左边为“传送源区域”，右边为“传送目标区域”。每一区域内又可以分为：

(1)“工程区域”——该区域的程序在计算机上。

(2) 机器人控制器区域。

(3) 存储在计算机其他文件夹中的程序。

选择某个区域，某个区域内的“程序”就以“一览表”的形式显示出来。对程序的复制、移动、删除、重新命名、比较等操作就可以在以上三个区域内互相进行。

如果左右区域相同，则可以进行复制、删除、更名、比较操作，但无法进行“移动”操作。

程序的复制、移动、删除、重新命名、比较等操作与一般软件相同，根据提示框就可以操作。

2. 保护的设定

保护功能是指对于被保护的文件，不允许进行移动、删除、名字变更等操作。保护功能仅对机器人控制器内的程序有效。

操作方法：选择要进行保护操作的程序。能够同时选择多个程序。左右两边的列表都

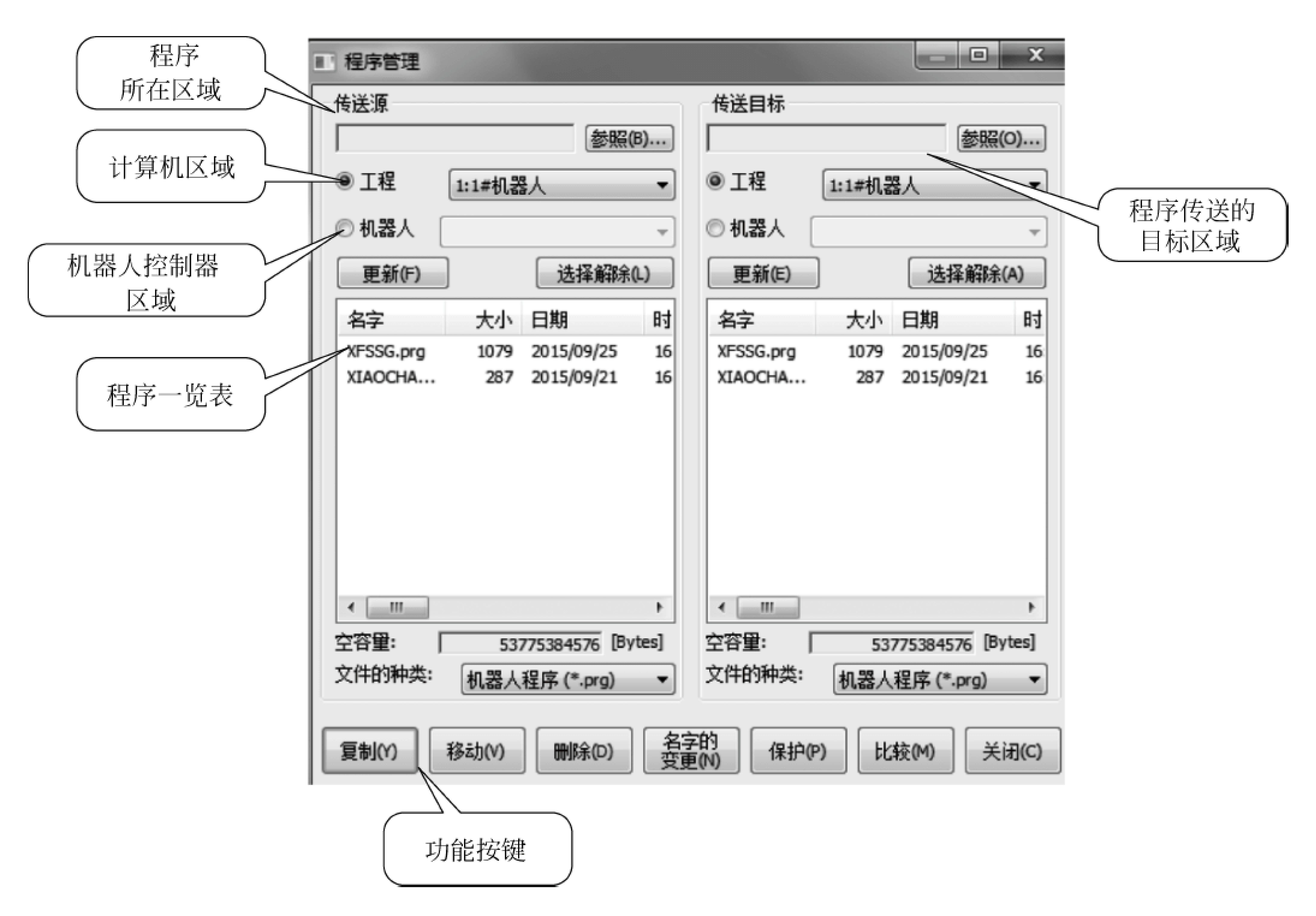


图 30-12　程序管理的区域及功能

能选择。单击“保护”按钮，在“保护设定”对话框中设定后，执行保护操作。

30.2.3　样条曲线的编制和保存

1. 编制样条曲线

右击“工程树”中的“在线”→“样条”，选择“新建”命令，如图 30-13 所示。

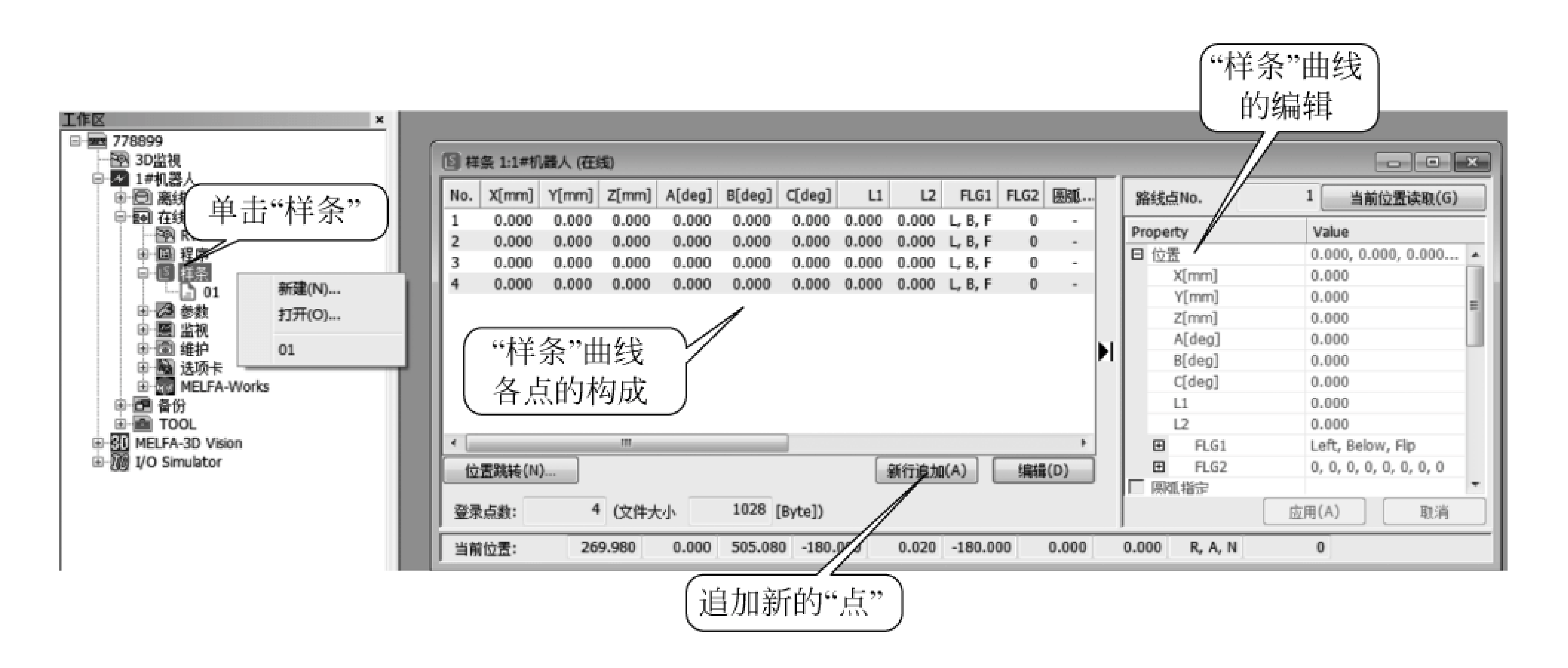


图 30-13　样条曲线的编辑窗口

由于样条曲线是由密集的“点”构成的，所以在如图 30-13 所示的窗口中，各“点”按表格排列，通过单击“新行追加”按钮可以追加新的“点”。在图 30-13 的右侧是对“位置点”的编



辑框，可以使用示教单元移动机器人通过读取“当前位置”获得新的“位置点”，也可以通过计算直接编辑位置点。

2. 保存

当样条曲线编制完成后，需要保存该文件，操作方法是单击“文件”→“保存”，该样条曲线文件就被保存。图 30-14 是“样条文件的保存”对话框。图 30-15 显示了已经制作保存的样条曲线名称、数量。



图 30-14　样条曲线的保存

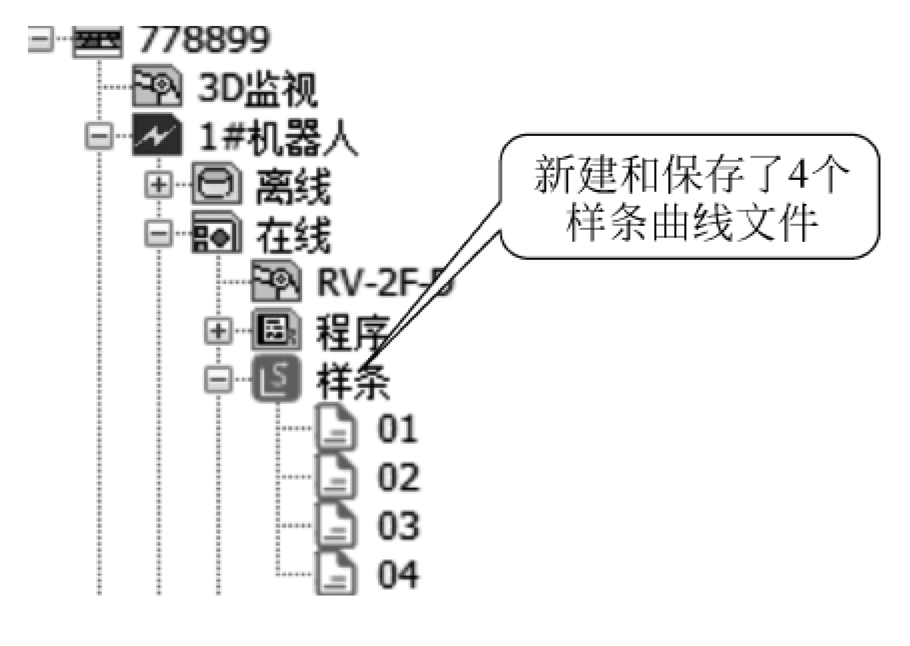


图 30-15　样条曲线的显示

在加工程序中使用 MVSPL 指令直接可以调用 ** 号样条曲线。这对于特殊运行轨迹的处理是很有帮助的。

30.2.4　程序的调试

1. 进入调试状态

从工程树的“在线”→“程序”中选择程序，单击鼠标右键，从弹出菜单中单击“调试状态下打开”，弹出如图 30-16 所示窗口。

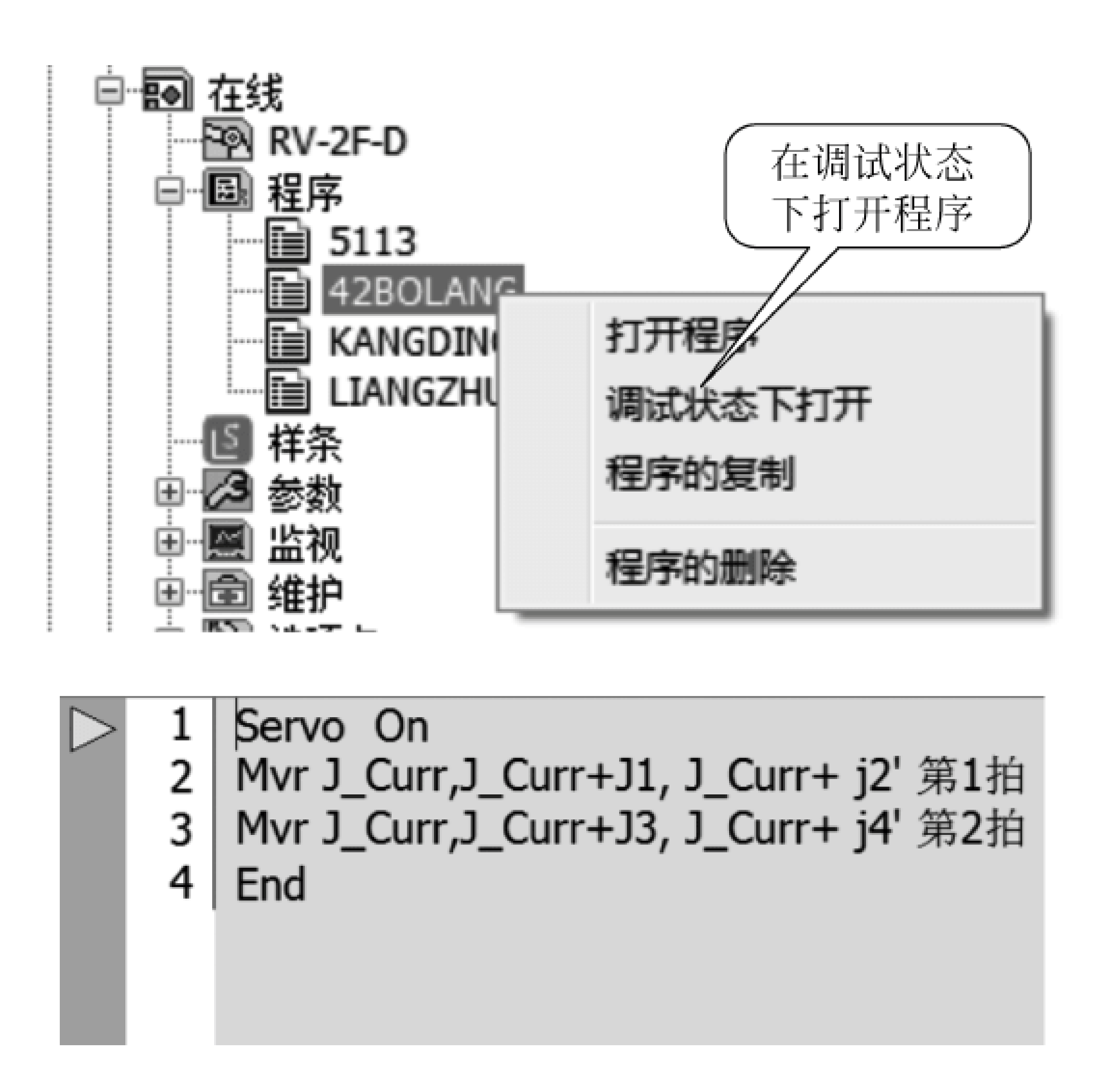


图 30-16　调试状态窗口

2. 调试状态下的程序编辑

调试状态下,通过菜单栏的"编辑"→"命令行编辑-在线"或"命令行插入-在线"或"命令行删除-在线"选项来编辑、插入和删除相关指令,如图 30-17 所示。

位置变量可以和通常状态一样进行编辑。



图 30-17　调试状态下的程序编辑

3. 单步执行

如图 30-18 所示,单击"操作面板"上的"前进""后退"按键,可以一行一行地执行程序。"继续"执行是使程序从"当前行"开始执行。

4. 操作面板上各按键和显示器上的功能

1）状态

显示控制器的任务区的状态,显示"待机中""可选择程序状态"。

2）OVRD

显示和设定速度比率。

3）跳转

可跳转到指定的程序行号。

4）停止

停止程序。

5）单步执行

一行一行执行指定的程序。单击"前进"按钮,执行当前行。单击"后退"按钮,执行上一

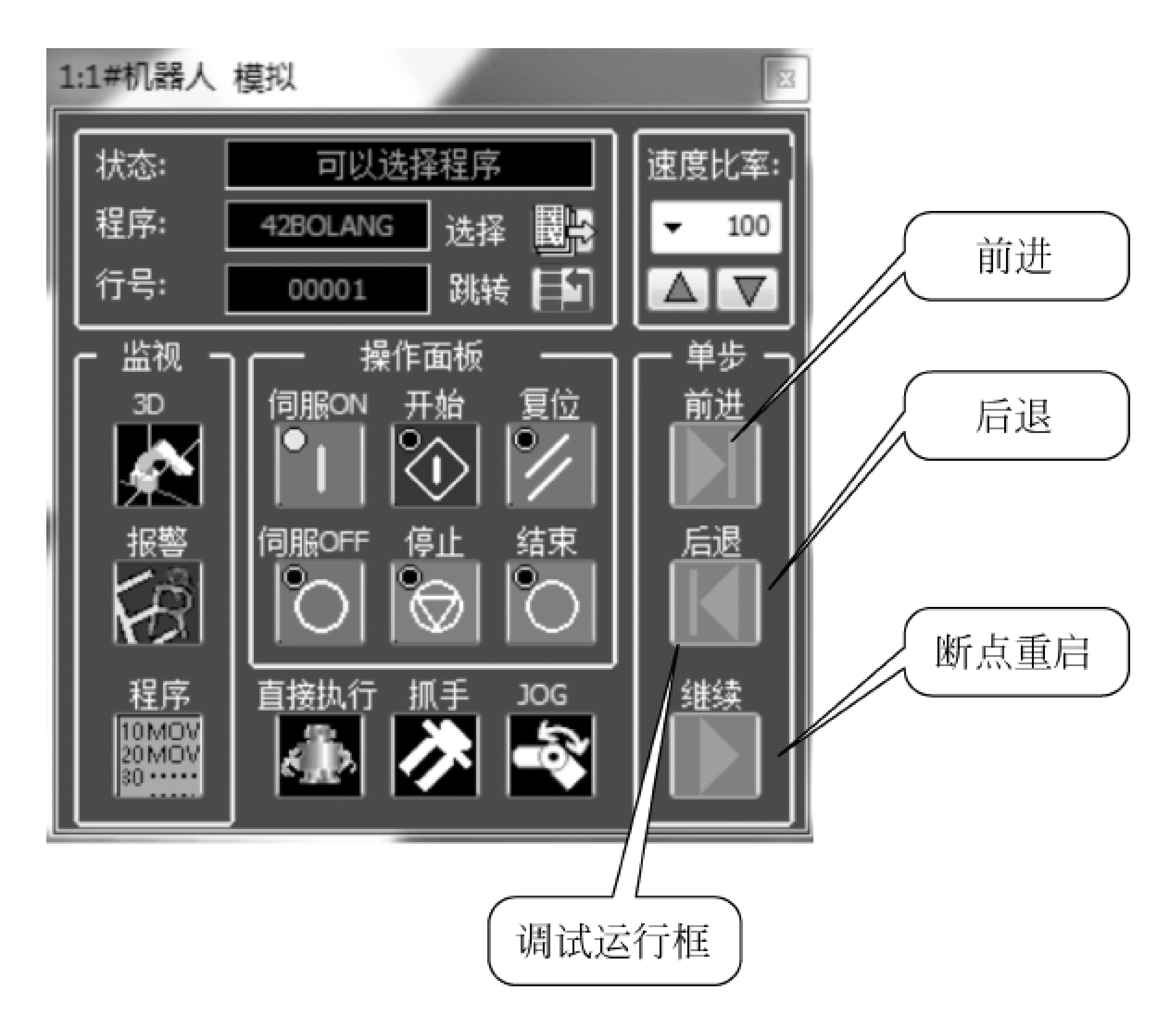


图 30-18 操作面板的各调试按键功能

行程序。

6）继续执行

程序从当前行开始继续执行。

7）伺服 ON/OFF

伺服 ON/OFF。

8）复位

复位当前程序及报警状态。可选择新的程序。

9）直接执行

和机器人程序无关，可以执行任意的指令。

10）3D 监视

显示机器人的 3D 监视。

5. 断点设置

在调试状态下可以对程序设定“断点”。所谓“断点”功能是指设置一个“停止位置”，程序运行到此位置就停止。

在调试状态下单步执行以及连续执行时，会在设定的“断点程序行”停止执行程序。停止后，再启动又可以继续单步执行。

断点最多可设定 128 个，程序关闭后全部解除。断点有以下两种。

(1) 持续断点：即便停止以后，断点仍被保存。

(2) 临时断点：停止后，断点会在停止的同时被自动解除。

断点的设置如图 30-19 所示。

6. 直接位置跳转

位置跳转功能是指选择某个位置点后直接运动到该位置点。

位置跳转的操作方法如下，如图 30-20 所示。

(1)（在有多个机器人的情况下）选择需要使其动作的机器人。

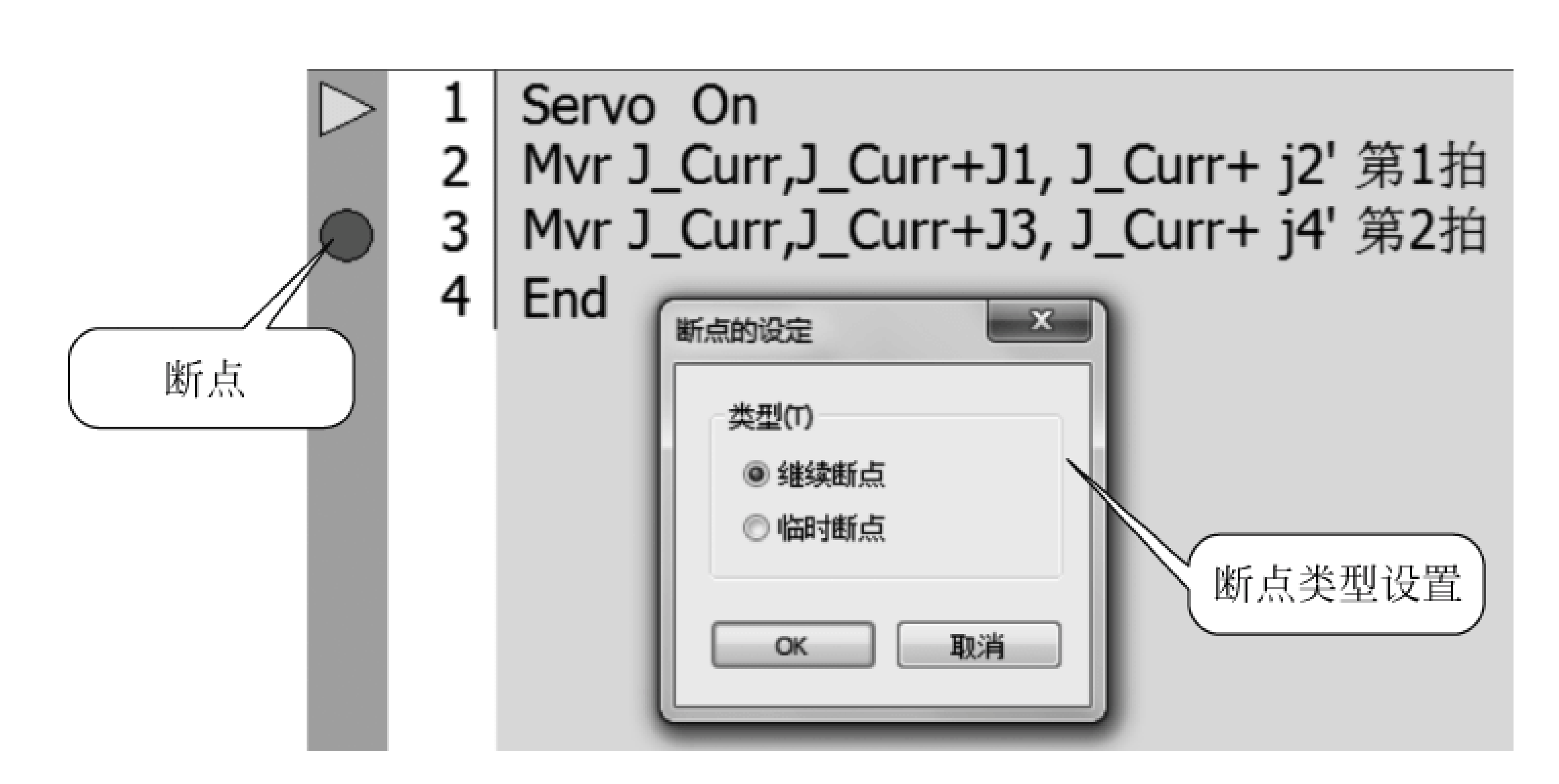
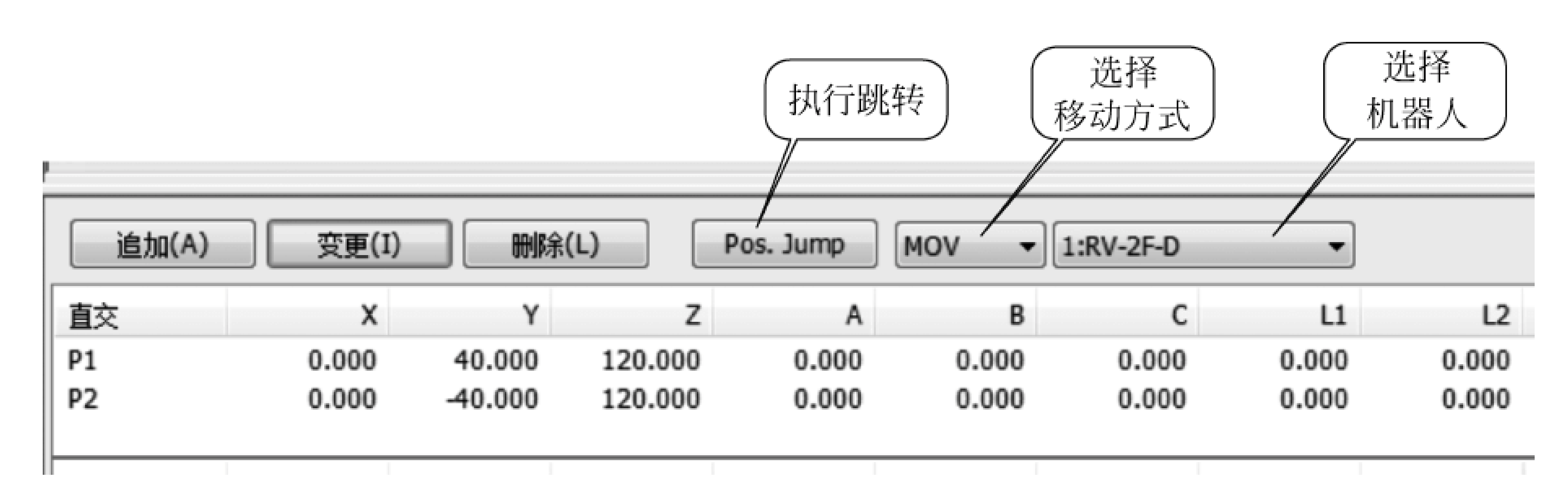


图 30-19　断点的设置

(2) 选择移动方法(MOV：关节插补移动，MVS：直线插补移动)。

(3) 选择要移动的位置点。



直交	X	Y	Z	A	B	C	L1	L2
P1	0.000	40.000	120.000	0.000	0.000	0.000	0.000	0.000
P2	0.000	-40.000	120.000	0.000	0.000	0.000	0.000	0.000

图 30-20　位置跳转的操作方法

(4) 单击 Pos. Jump 按钮。

在实际使机器人动作的情况下，会显示提醒注意的警告。

7. 退出调试状态

要结束“调试状态”，单击程序框中的“关闭”图标即可，如图 30-21 所示。

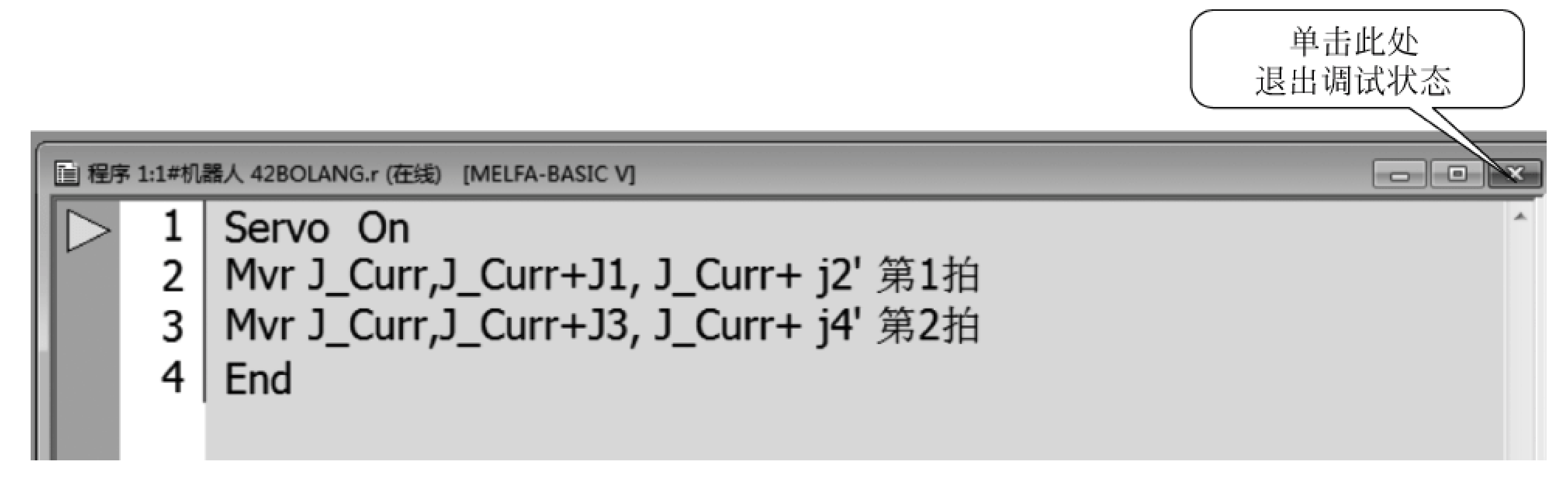


图 30-21　关闭调试状态

30.3　参数设置

参数设置是本软件的重要功能。可以在软件上或示教单元上对机器人设置参数。各参数的功能已经在第 8 章中做了详细说明，在对参数有了正确理解后用本软件可以快速方便



地设置参数。

30.3.1　使用参数一览表

单击工程树“离线”→“参数”→“参数一览”，弹出如图 30-22 所示的“参数一览表”。“参数一览表”按参数的英文字母顺序排列，双击需要设置的参数后，弹出该参数的设置框，如图 30-23 所示，根据需要进行设置。



图 30-22　参数一览表



图 30-23　参数设置框

使用参数一览表的好处是可以快速查找和设置参数，特别是知道参数的英文名称时可以快速设置。

30.3.2　按功能分类设置参数

1. 参数分类

为了按同一类功能设置参数，本软件还提供了按参数功能分块设置的方法。这种方法很实用，在实际调试设备时通常使用这一方法。本软件将参数分为以下几个大类。

（1）动作参数；

（2）程序参数；

（3）信号参数；

（4）通信参数；

（5）现场网络参数。

每一大类又分为若干小类。

2. 动作参数

1）动作参数分类

单击“动作参数”，展开如图 30-24 所示列表：这是动作参数内的各小分类，根据需要选择。

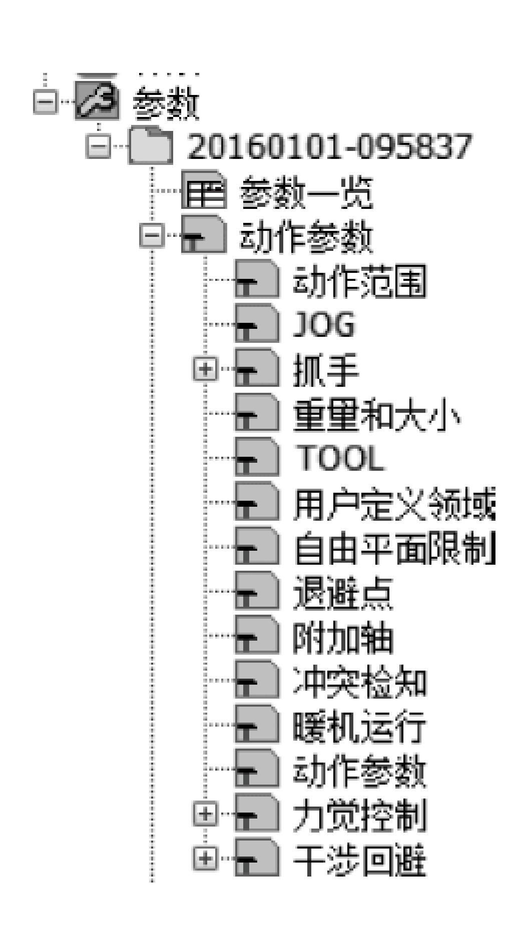


图 30-24 动作参数分类

2）设置具体参数

操作方法如下。

单击“离线”→“参数”→“动作参数”→“动作范围”，弹出如图 30-25 所示的“动作范围设置”框，在这一“动作范围设置”框内，可以设置各轴的“关节动作范围”、在“直交坐标系内的动作范围”等内容，既明确又快捷方便。

3. 程序参数

1）程序参数分类

单击“程序参数”，展开如图 30-26 所示列表：这是程序参数内的各小分类，根据需要选择。

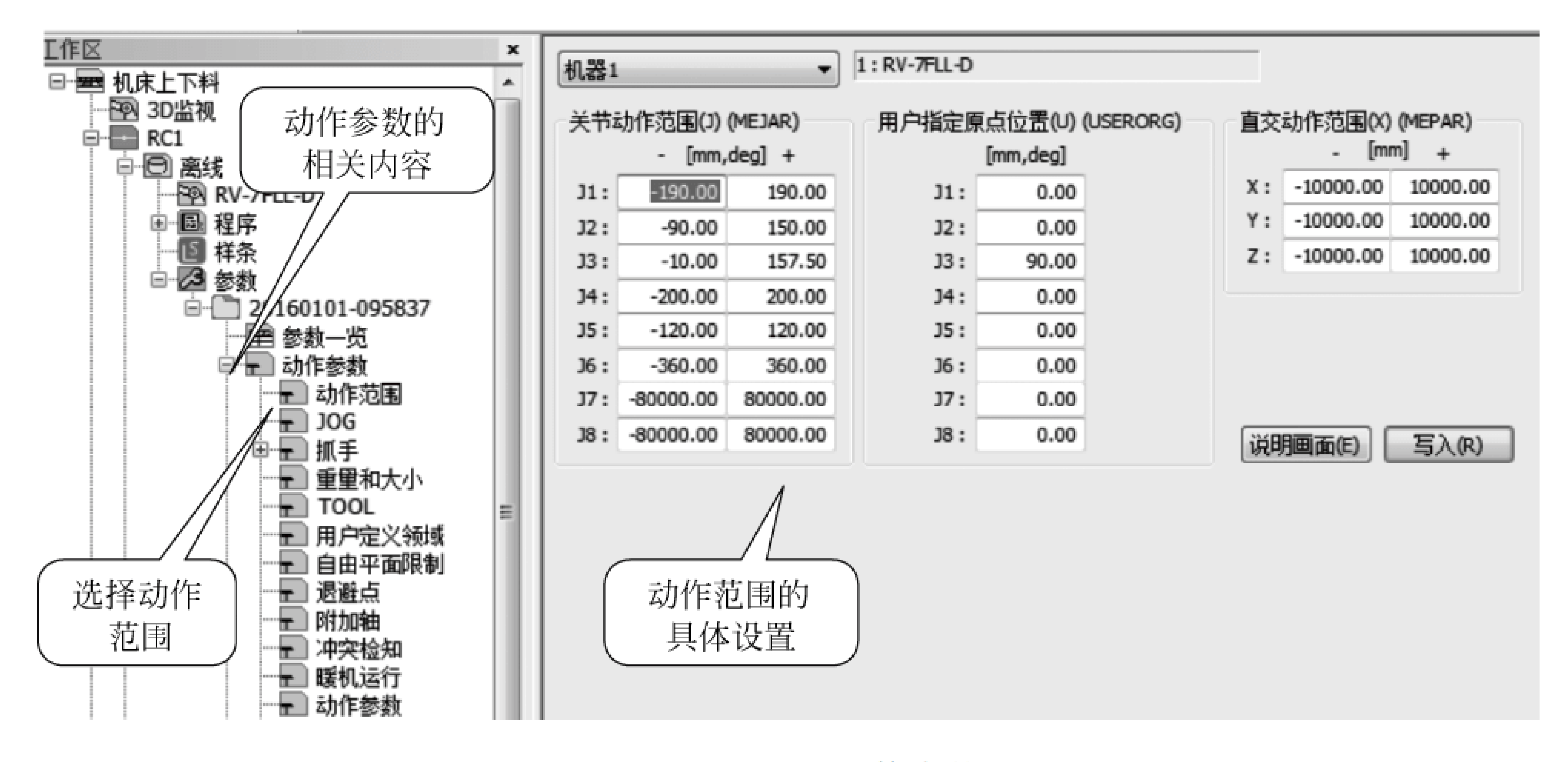


图 30-25 设置具体参数

2）设置具体参数

操作方法如下。

单击“离线”→“参数”→“程序参数”→“插槽表”，弹出如图 30-27 所示的“插槽表”，在插槽表设置框内，可以设置需要预运行的程序。

4. 信号参数

1）信号参数分类

单击“信号参数”，展开如图 30-28 所示列表。这是信号参数内的各小分类，根据需要选择。

2）设置具体参数

操作方法如下。

单击“离线”→“参数”→“信号参数”→“专用输入输出信号分配”→“通用 1”，弹出如



图 30-26　程序参数分类

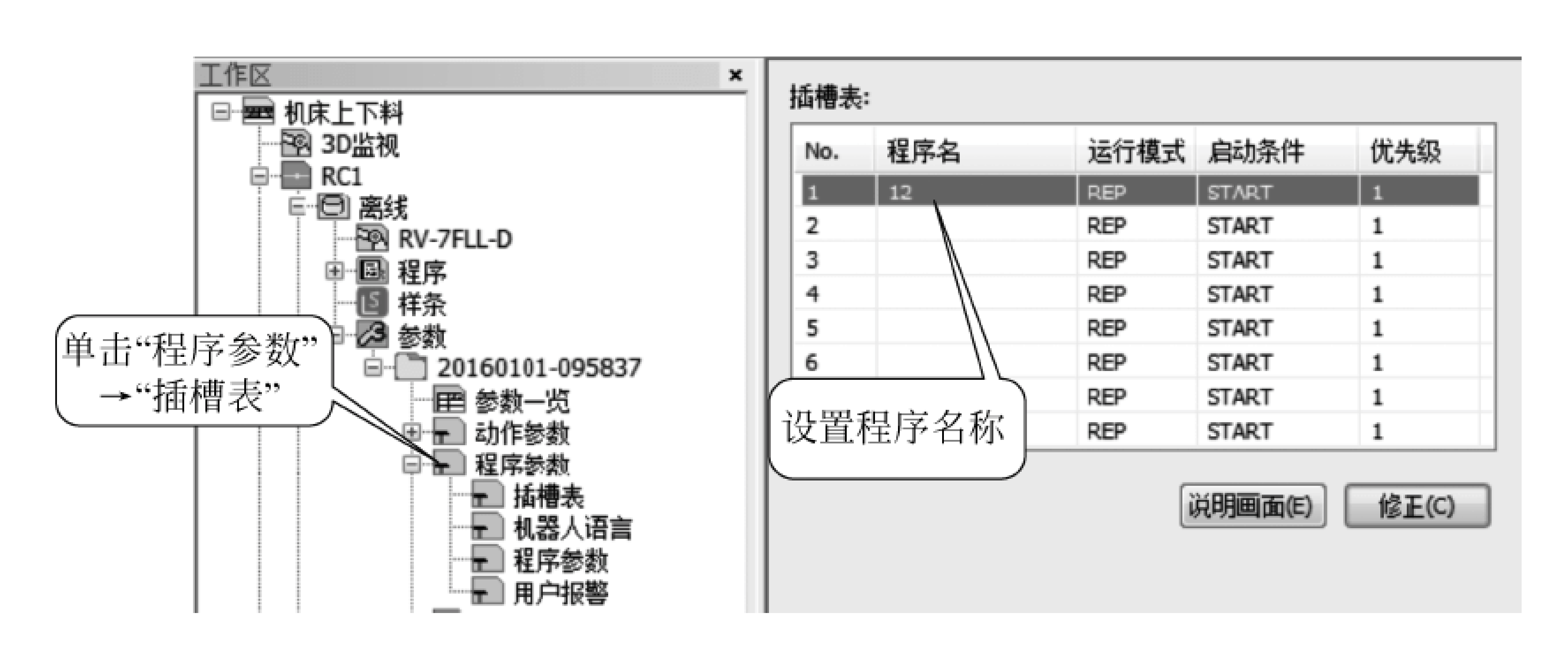


图 30-27　设置具体参数

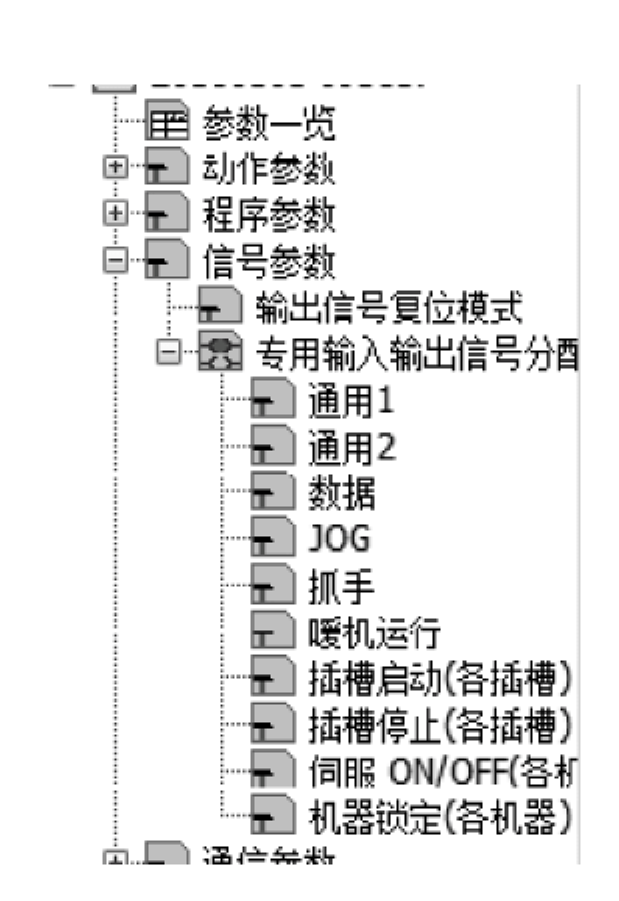


图 30-28　信号参数分类

图 30-29 所示的"专用输入输出信号设置"框，在"专用输入输出信号设置"框内，可以设置相关的输入输出信号。

5. 通信参数

1）通信参数分类

单击"通信参数"，展开如图 30-30 所示列表。这是通信参数内的各小分类，根据需要选择。

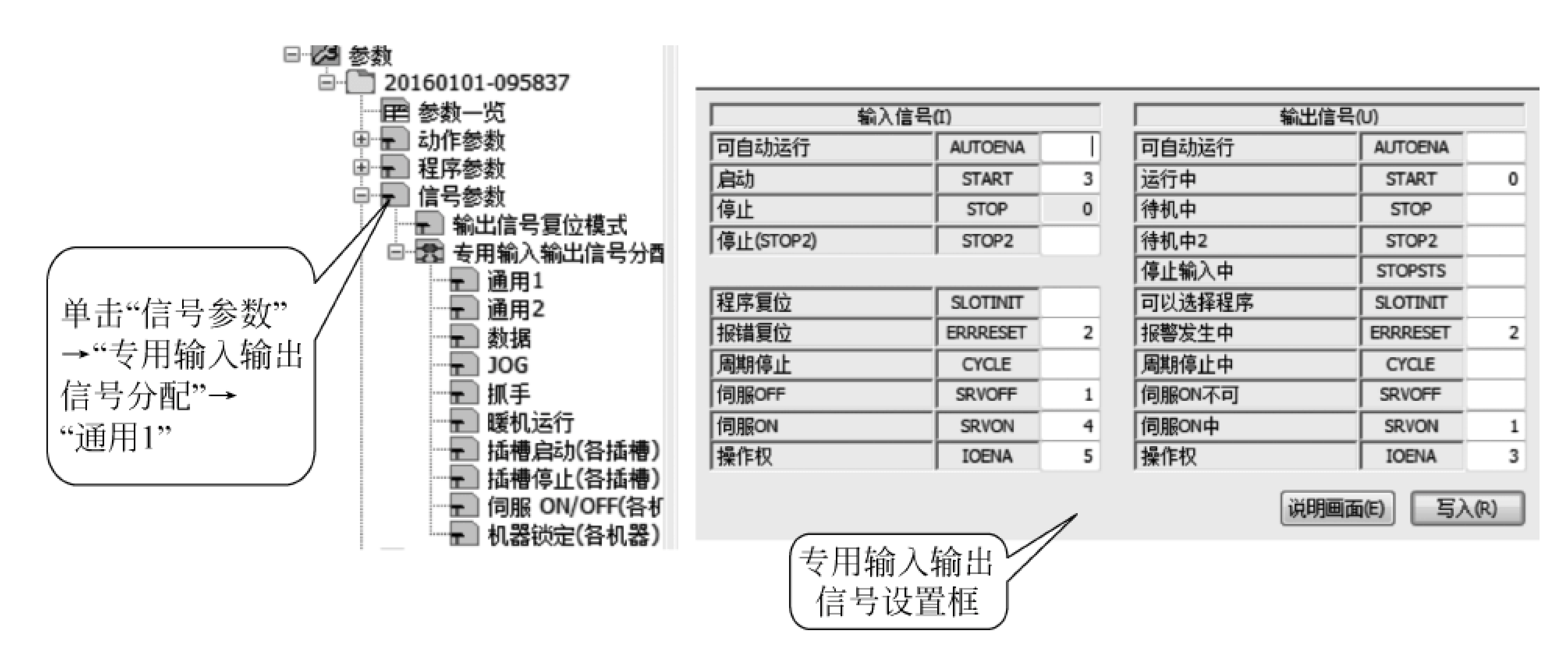


图 30-29 设置具体参数

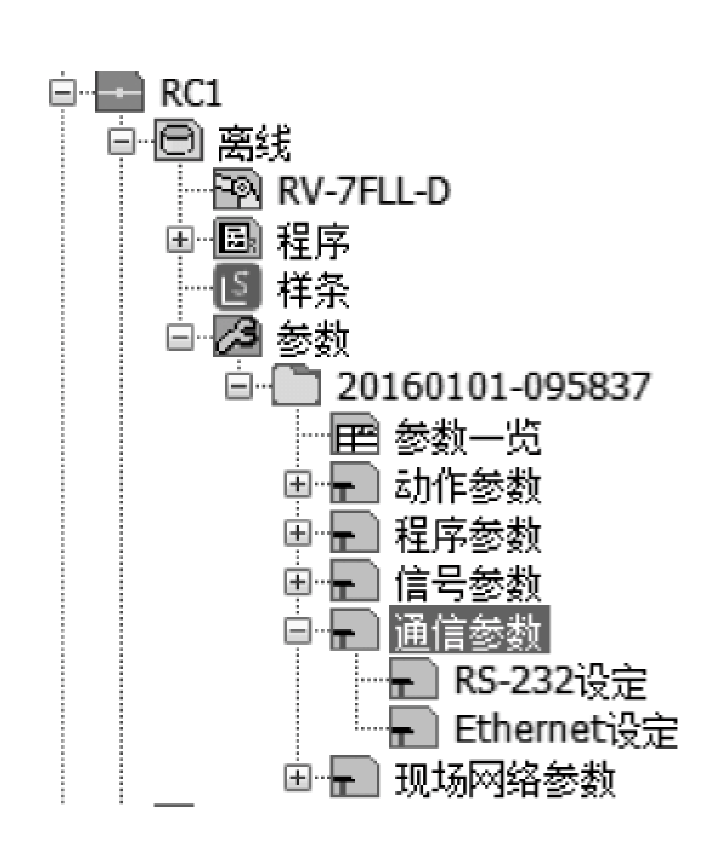


图 30-30 通信参数分类

2）设置具体参数

操作方法如下。

单击“离线”→“参数”→“通信参数”→Ethernet，弹出如图 30-31 所示的“以太网通信参数设置”框，在“以太网通信参数设置”框内，可以设置相关的通信参数。

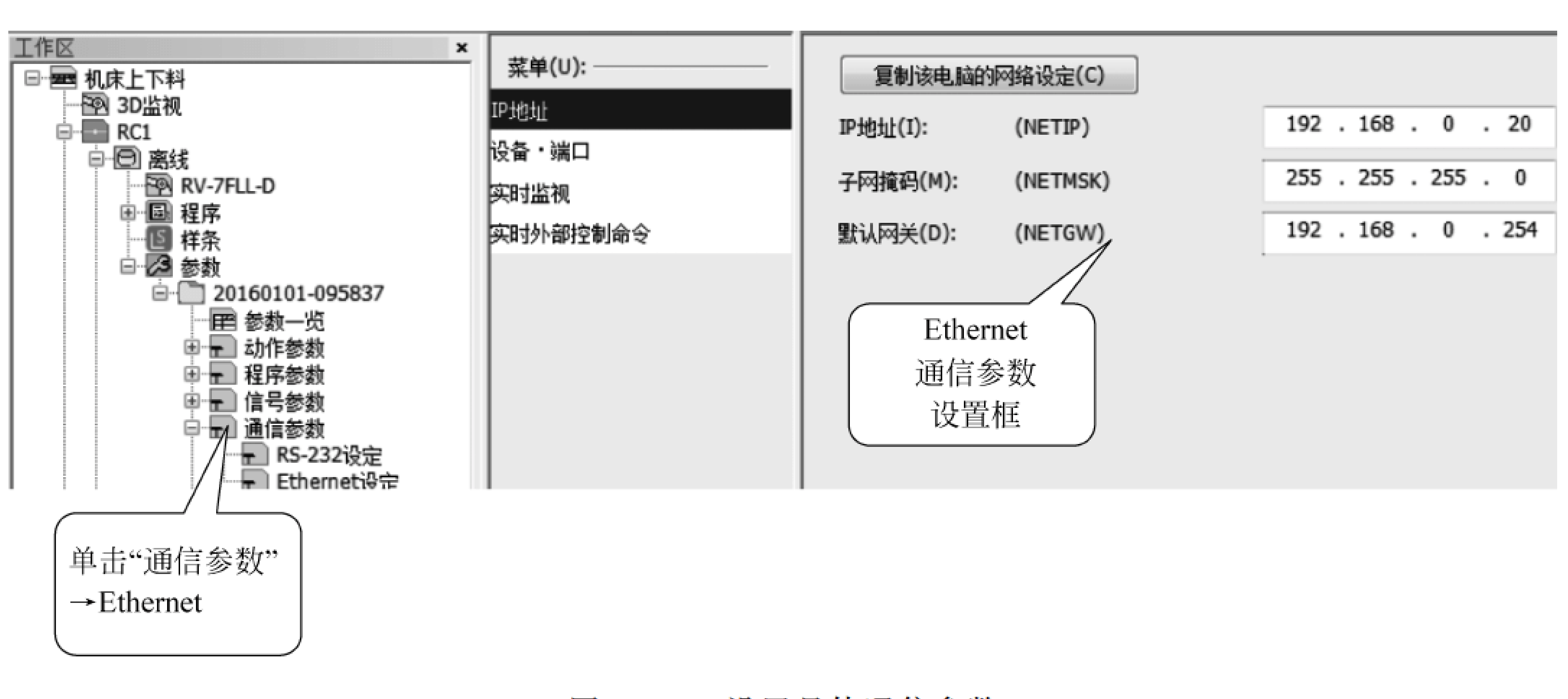


图 30-31 设置具体通信参数

6. 现场网络参数

单击“现场网络参数”，展开如图 30-32 所示列表。这是现场网络参数内的各小分类，根据需要选择设置。

图 30-32　现场网络参数分类

30.4　机器人工作状态监视

30.4.1　动作监视

1. 任务区状态监视

监视对象：任务区的工作状态，即显示任务区（SLOT）是否可以写入新的程序。如果该任务区内的程序正在运行，就不可写入新的程序。

单击“监视”→“动作监视”→“插槽状态”，弹出“插槽状态监视”框。

“插槽（SLOT）”就是“任务区”，如图 30-33 所示。

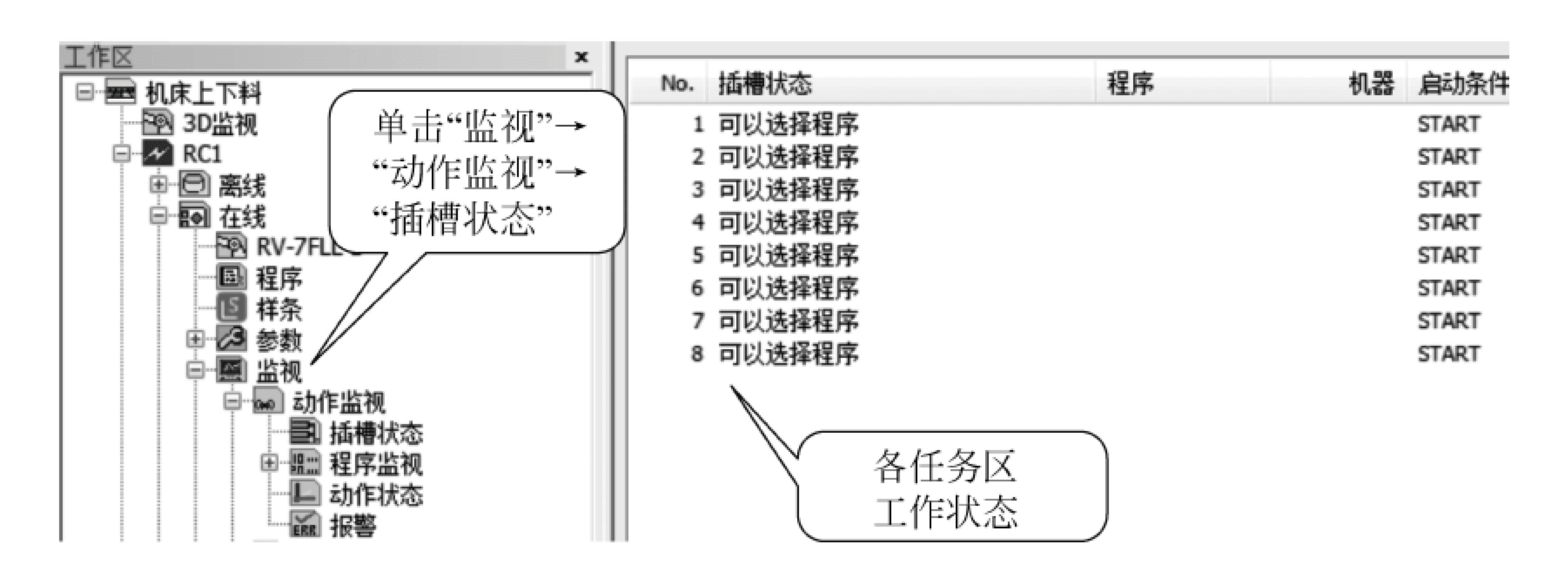


图 30-33　插槽状态监视框

2. 程序监视

监视对象：任务区内正在运行程序的工作状态，即正在运行的程序行。

单击“监视”→“动作监视”→“程序监视”，弹出“程序监视”框，如图 30-34 所示。

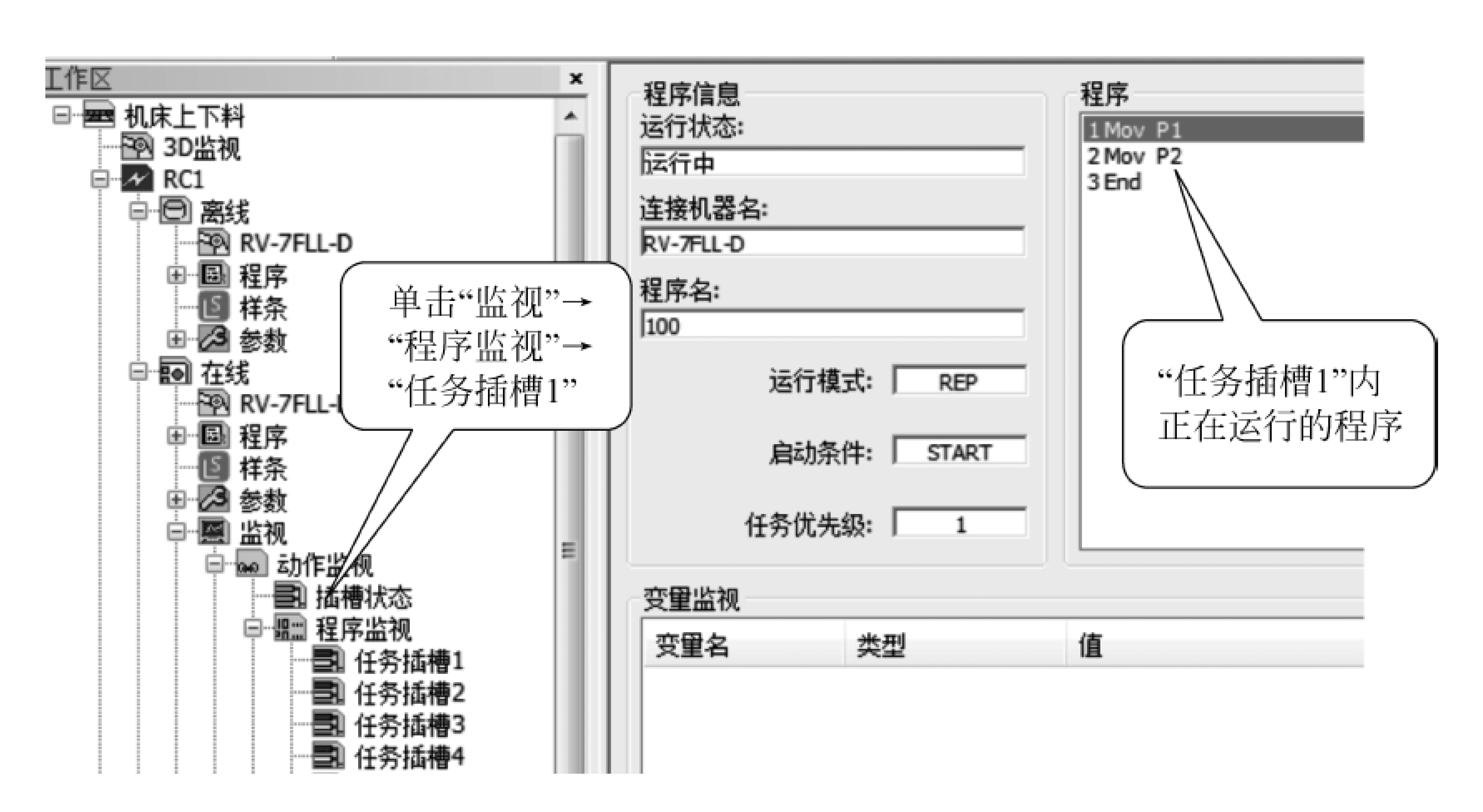


图 30-34 “程序监视”框

3. 动作状态监视

监视对象如下。

(1) 直交坐标系中的当前位置;

(2) 关节坐标系中的当前位置;

(3) 抓手 ON/OFF 状态;

(4) 当前速度;

(5) 伺服 ON/OFF 状态。

单击“监视”→“动作监视”→“动作状态”,弹出“动作状态”框,如图 30-35 所示。

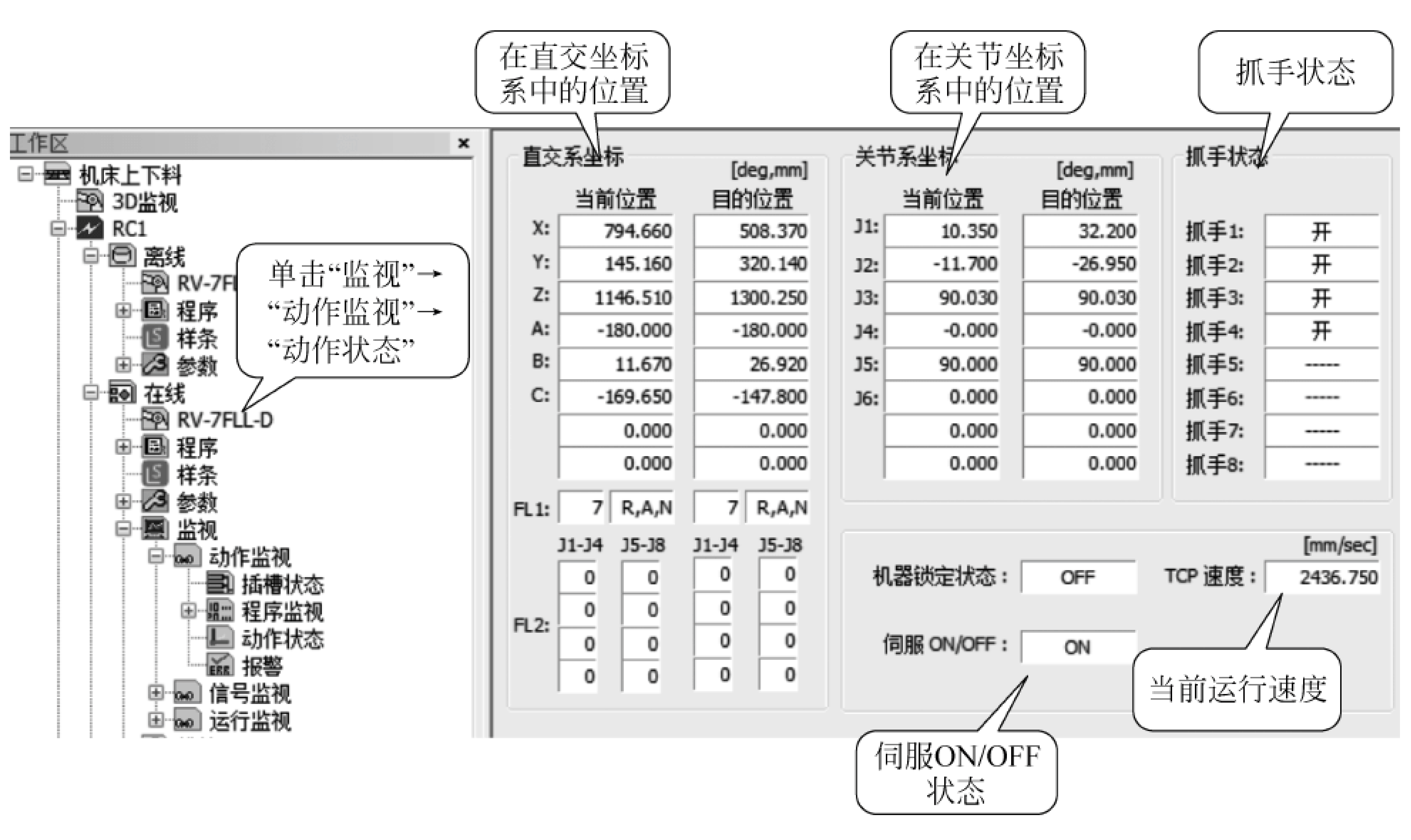


图 30-35 “动作状态”框

4. 报警内容监视

单击“监视”→“动作监视”→“报警”,弹出“报警”框,如图 30-36 所示。在“报警”框内显示报警号、报警信息、报警时间等内容。



图 30-36 “报警”框

30.4.2 信号监视

1. 通用信号的监视和强制输入输出

功能：用于监视输入输出信号的 ON/OFF 状态。

单击“监视”→“信号监视”→“通用信号”，弹出“通用信号”窗口如图 30-37 所示。在“通用信号”窗口除了监视当前输入输出信号的 ON/OFF 状态以外，还可以：

(1) 模拟输入信号；

(2) 设置监视信号的范围；

(3) 强制输出信号 ON/OFF。

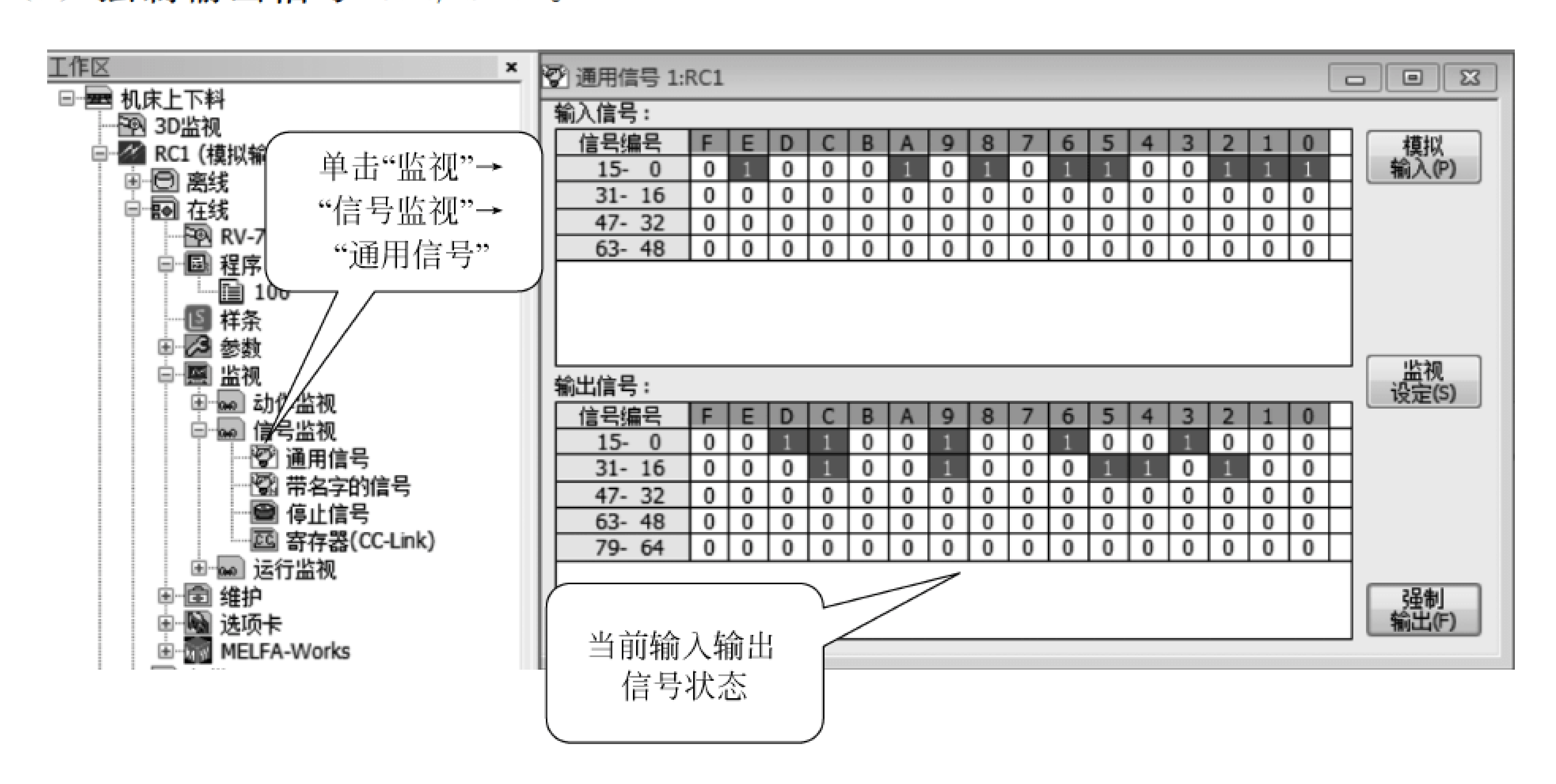


图 30-37 “通用信号”窗口的监视状态

2. 对已经命名的输入输出信号监视

功能：用于监视已经命名的输入输出信号的 ON/OFF 状态。

单击“监视”→“信号监视”→“带名字的信号”，弹出“带名字的信号”窗口，如图 30-38 所示。在“带名字的信号”窗口内可以监视已经命名的输入输出信号的 ON/OFF 状态。

3. 对停止信号以及急停信号监视

功能：用于监视停止信号以及急停信号的 ON/OFF 状态。

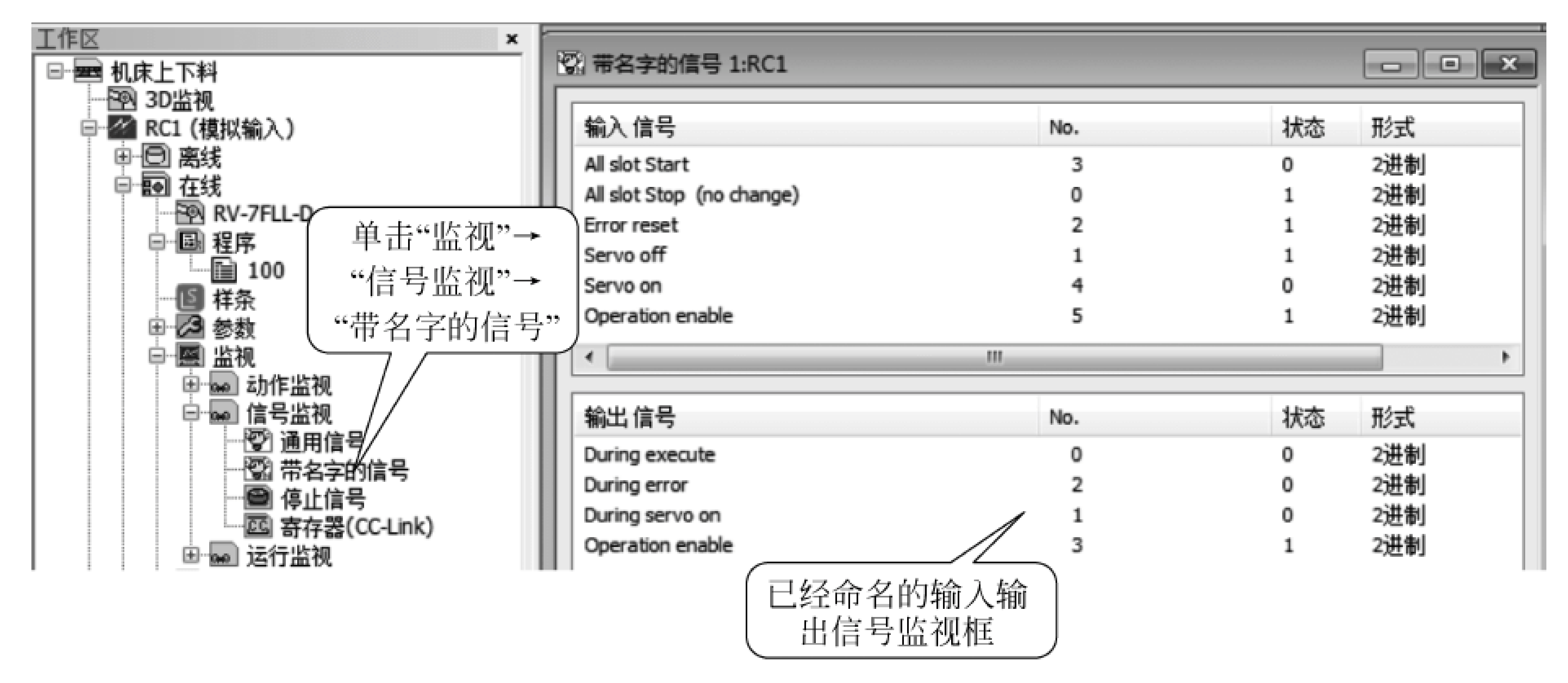


图 30-38 “带名字的信号”窗口内监视已经命名的输入输出信号的 ON/OFF 状态

单击“监视”→“信号监视”→“停止信号”，弹出“停止信号”窗口如图 30-39 所示。在“停止信号”窗口内可以监视停止信号以及急停信号的 ON/OFF 状态。

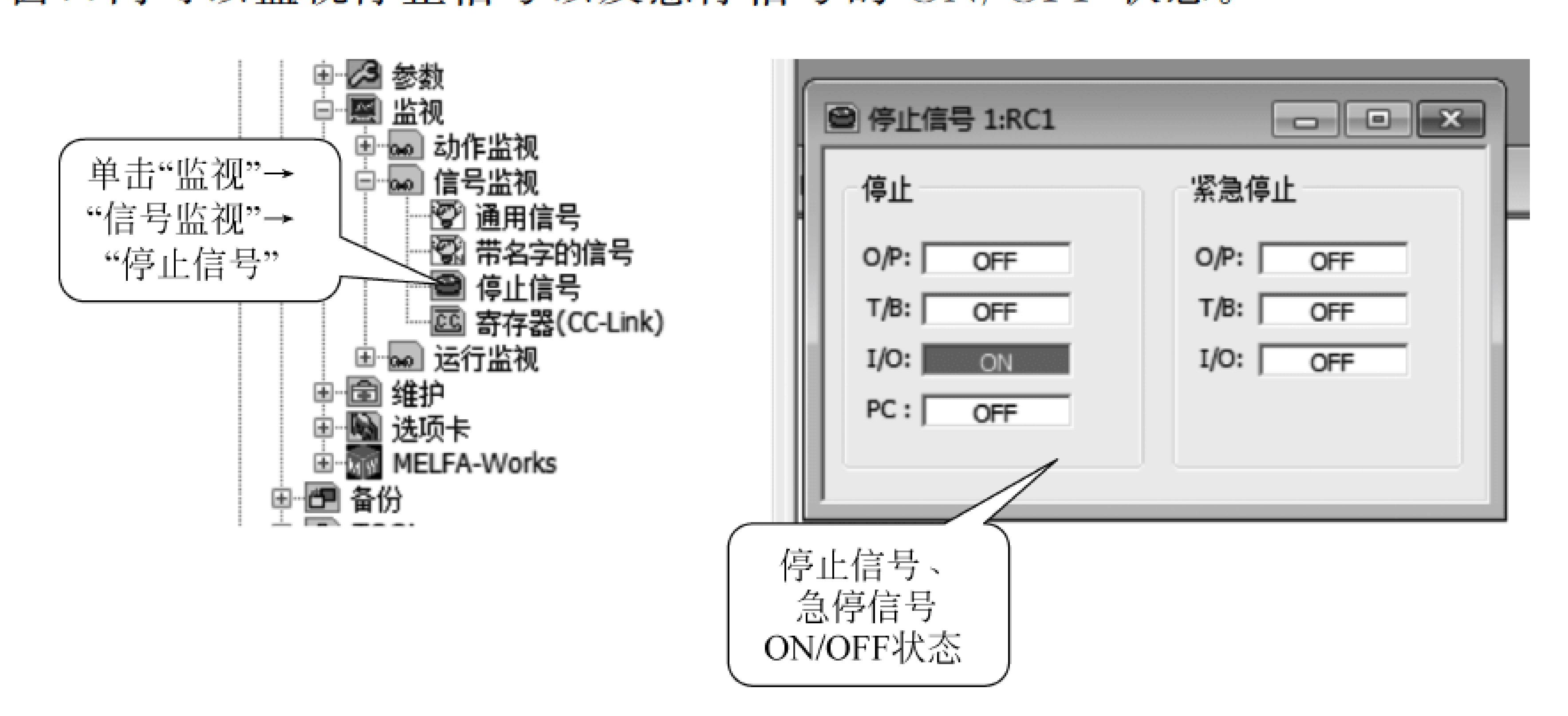


图 30-39 “停止信号”窗口内监视停止信号以及急停信号的 ON/OFF 状态

30.4.3 运行监视

功能：用于监视机器人系统的运行时间。

单击“监视”→“运行监视”→“运行时间”，弹出“运行时间”窗口如图 30-40 所示。在“运行时间”窗口内可以监视“电源 ON 时间”“运行时间”“伺服 ON 时间”等内容。

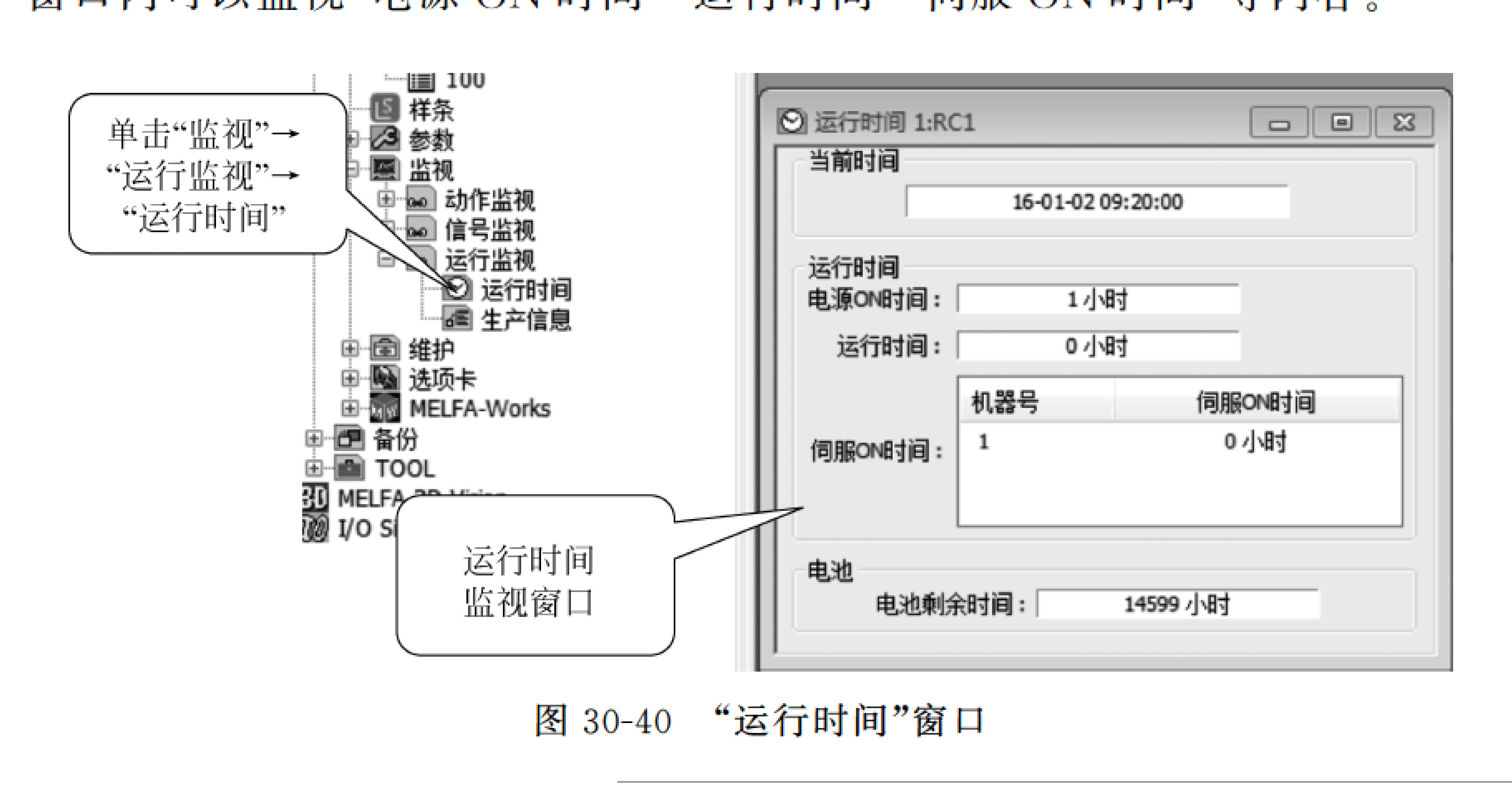


图 30-40 “运行时间”窗口

30.5 维　　护

30.5.1 原点设置

1. 设置方式

功能：进行原点设置和恢复。设置原点有6种方式，如图30-41所示。

(1) 原点数据输入方式；

(2) 机械限位器方式；

(3) 夹具方式；

(4) ABS原点方式；

(5) 用户原点方式；

(6) 原点参数备份方式。

单击"维护"→"原点数据"，弹出如图30-41所示"原点数据"设置窗口。

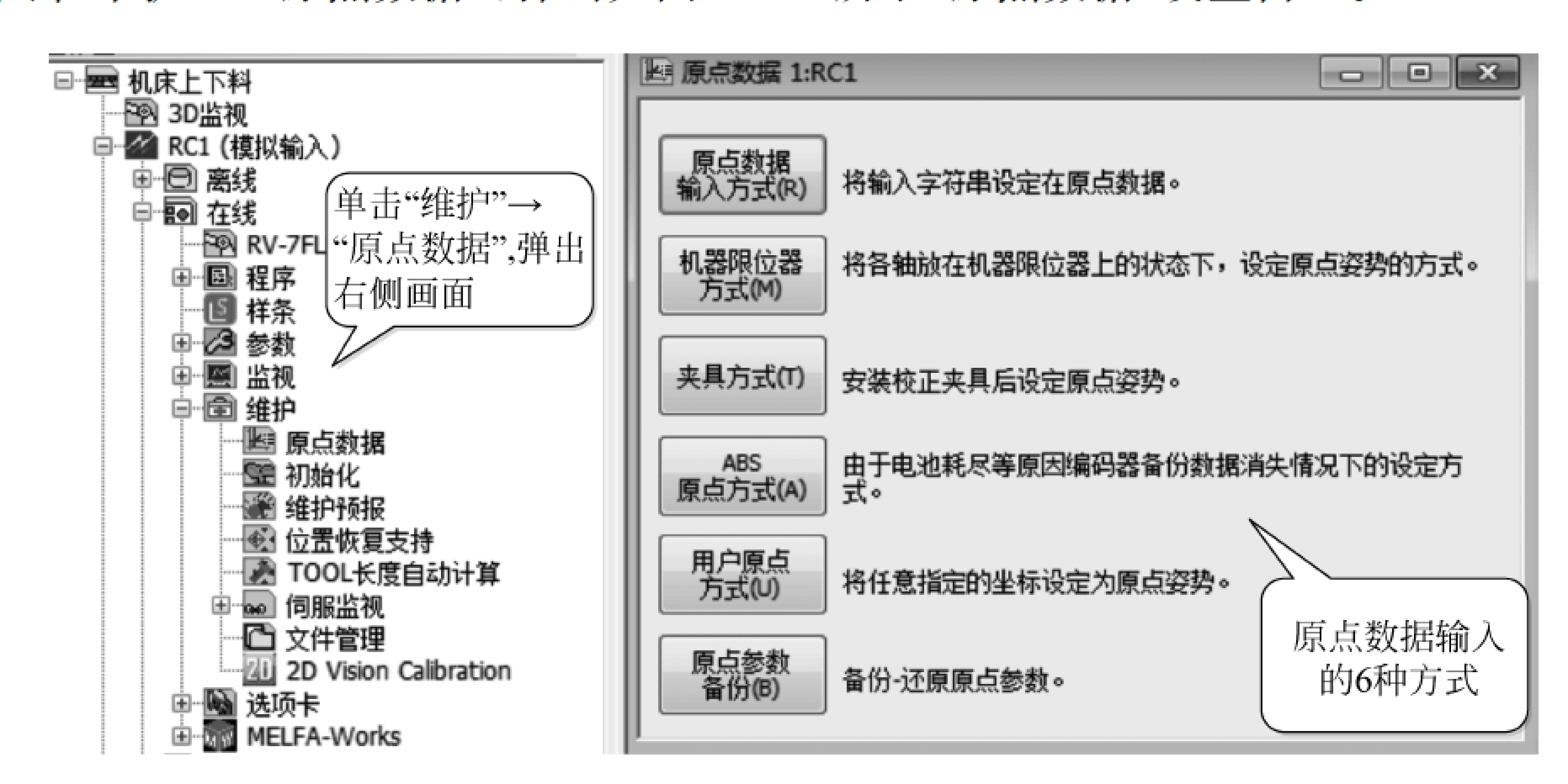


图30-41　原点数据设置窗口

2. 原点数据输入方式

原点数据输入方式——直接输入"字符串"。

功能：将出厂时厂家标定的原点写入控制器。出厂时，厂家已经标定了各轴的原点。并且作为随机文件提供给使用者。一方面使用者在使用前应该输入"原点文件"——原点文件中每一轴的原点是一"字符串"。使用者应该妥善保存"原点文件"。另一方面，如果原点数据丢失后，可以直接输入原点文件的字符串，以恢复原点。

本操作需要在联机状态下操作。单击"原点数据输入方式"，弹出如图30-42所示"原点数据"设定窗口，各按钮作用如下。

(1) 写入——将设置完毕的数据写入控制器。

(2) 保存文件——将当前原点数据保存到计算机中。

(3) 从文件读出——从计算机中读出"原点数据文件"。

(4) 更新——从控制器内读出的"原点数据"，显示最新的原点数据。

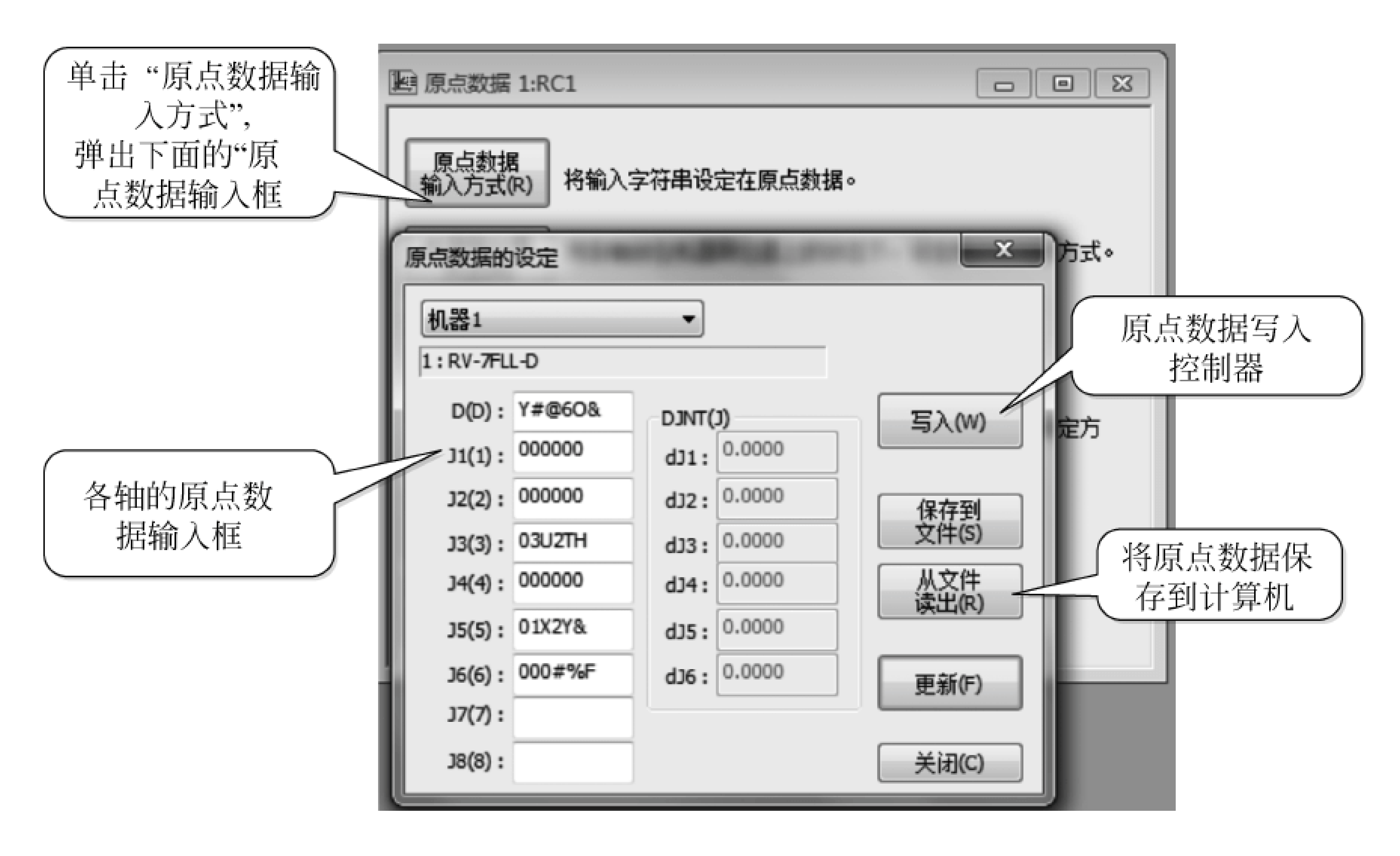


图 30-42　原点数据输入方式——直接输入"字符串"

3. 机械限位器方式

功能：以各轴机械限位器位置为原点位置。

操作方法：如图 30-43 所示。

(1) 单击原点数据画面中的"机器限位器方式"按钮，显示画面；

(2) 将机器人移动到机器限位器位置；

(3) 选中需要做原点设定的轴的复选框；

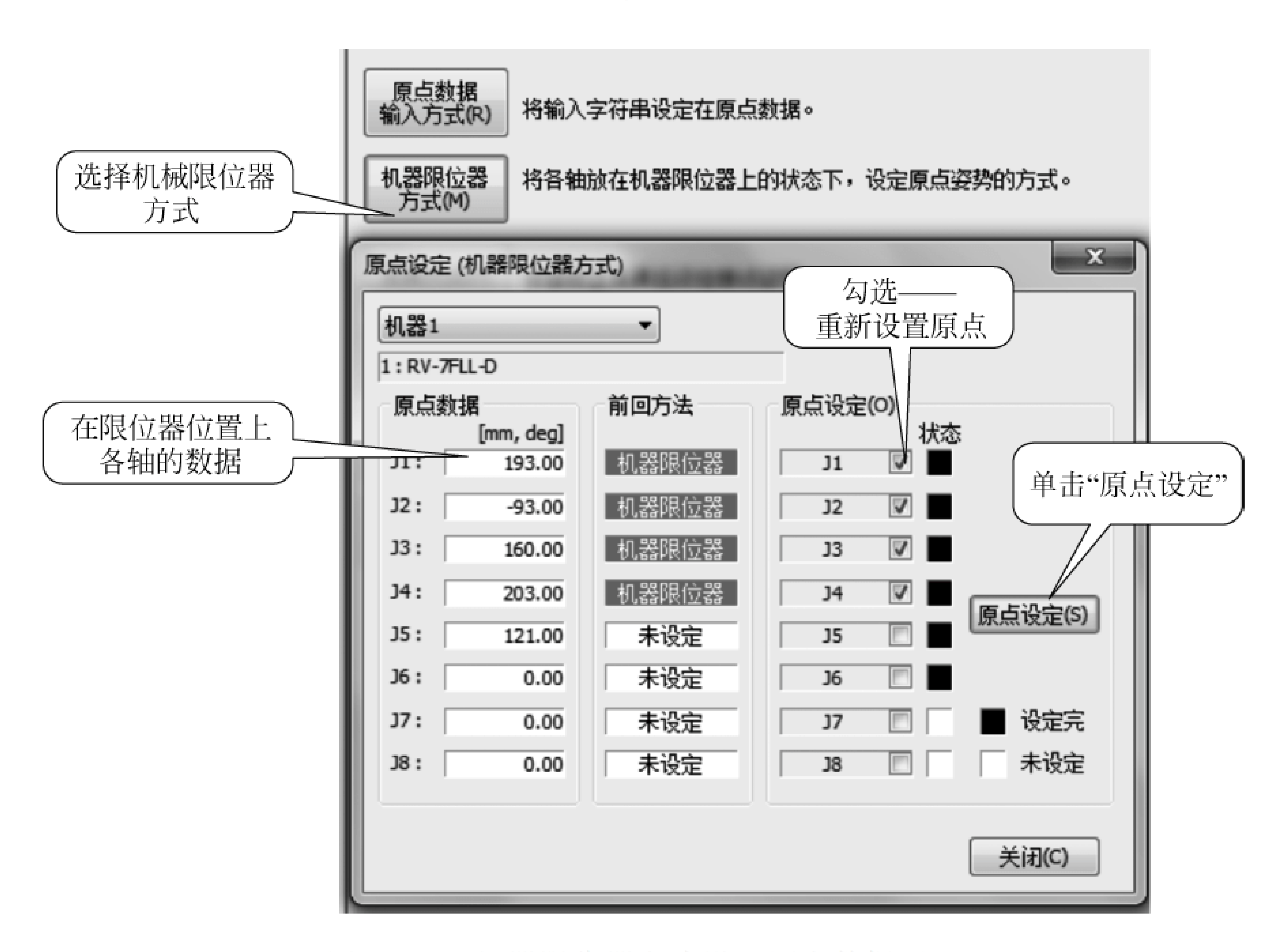


图 30-43　机器限位器方式设置原点数据画面

(4) 单击“原点设定”按钮(原点设置完成)。

图中前一次方法中,会显示前一次原点设定的方式。

4. 工具校准棒方式

功能:以“校正棒”校正各轴的位置,并将该位置设置为原点。

操作方法:如图30-44所示。

(1) 单击原点数据画面中的“夹具方式”按钮,显示画面如图30-44所示(夹具方式就是校正棒方式);

(2) 将机器人各轴移动到“校正棒”校正的各轴位置;

(3) 选中需要做原点设定的轴的复选框;

(4) 单击“原点设定”按钮(原点设置完成)。

图中前一次方法中,会显示前一次原点设定的方式。

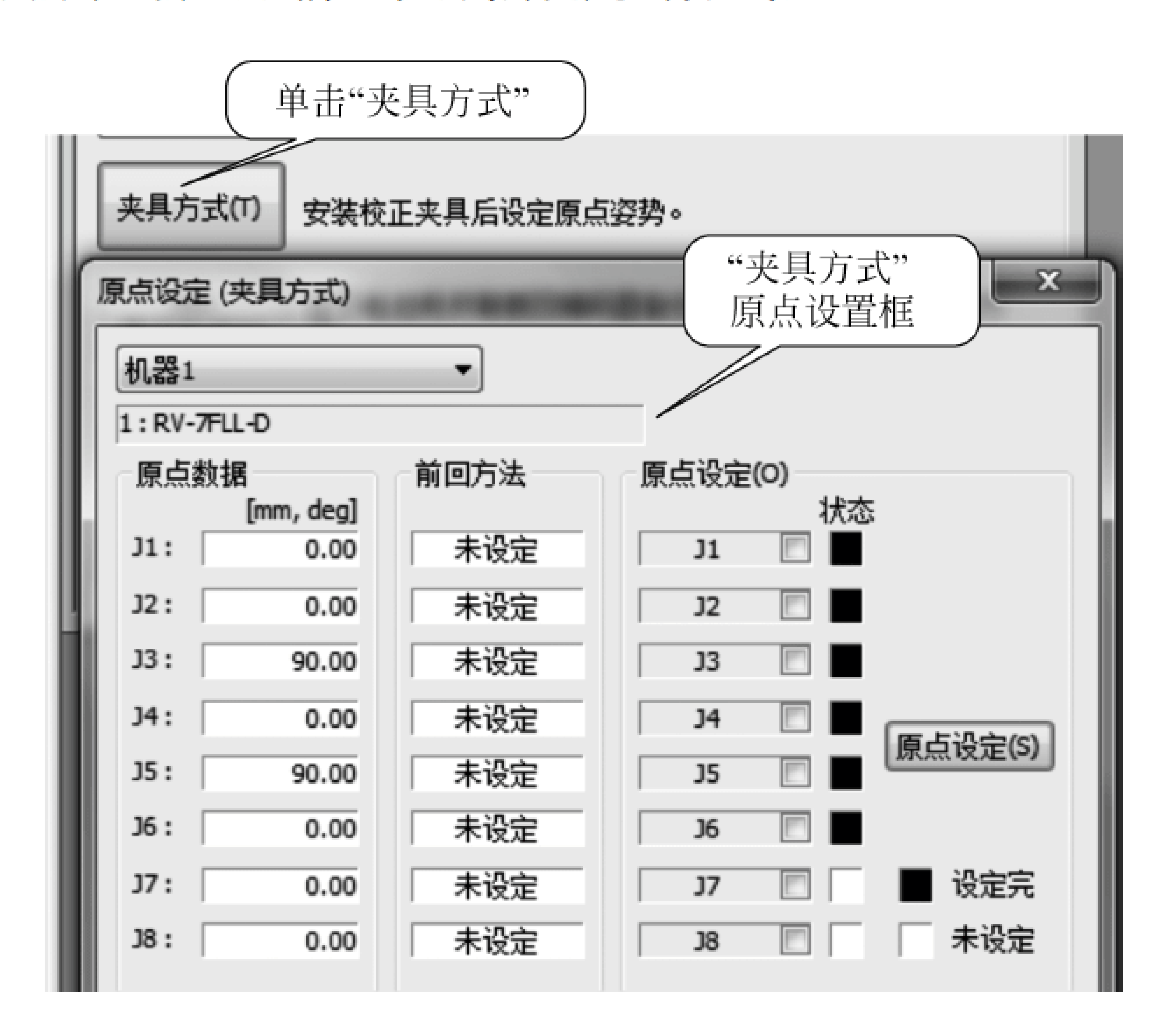


图30-44 夹具方式设置原点数据画面

5. ABS原点方式

功能:在机器人各轴位置都有一个三角符号“Δ”,将各轴的三角符号“Δ”与相邻轴的三角符号“Δ”对齐,此时各轴的位置就是“原点位置”。

操作方法:如图30-45所示。

(1) 单击原点数据画面中的“ABS方式”按钮,显示画面如图30-45所示;

(2) 将机器人各轴移动到三角符号“Δ”对齐的位置;

(3) 选中需要做原点设定的轴的复选框;

(4) 单击“原点设定”按钮(原点设置完成)。

图中前一次方法中,会显示前一次原点设定的方式。

6. 用户原点方式

功能:由用户自行定义机器人的任意位置为“原点位置”。

操作方法如下。

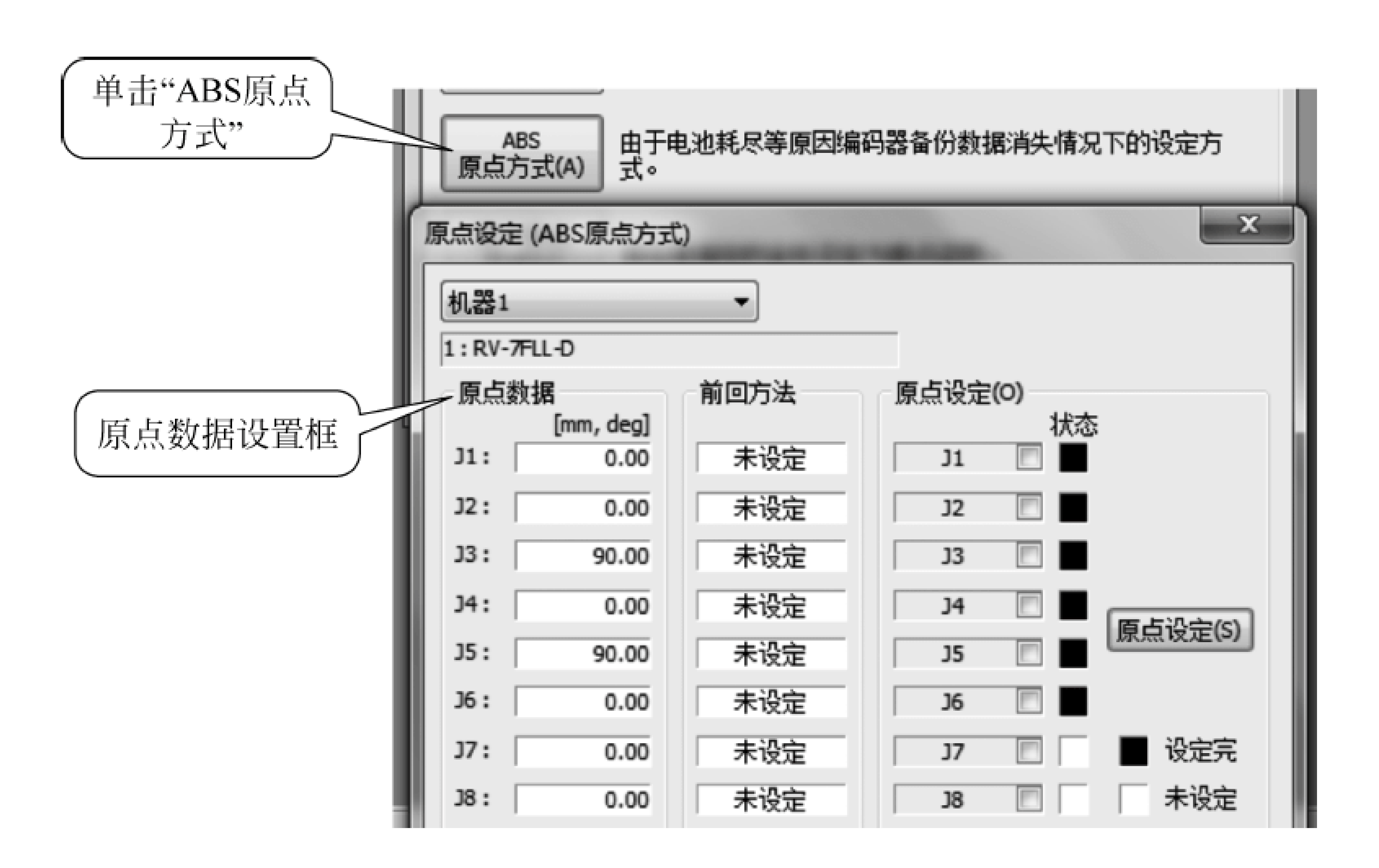


图 30-45　ABS 方式设置原点数据画面

（1）单击原点数据画面中的“用户原点”按钮，显示画面类似图 30-46；

（2）将机器人各轴移动到用户任意定义的原点位置；

（3）选中需要做原点设定的轴的复选框；

（4）单击“原点设定”按钮(原点设置完成)。

图中前一次方法中，会显示前一次原点设定的方式。

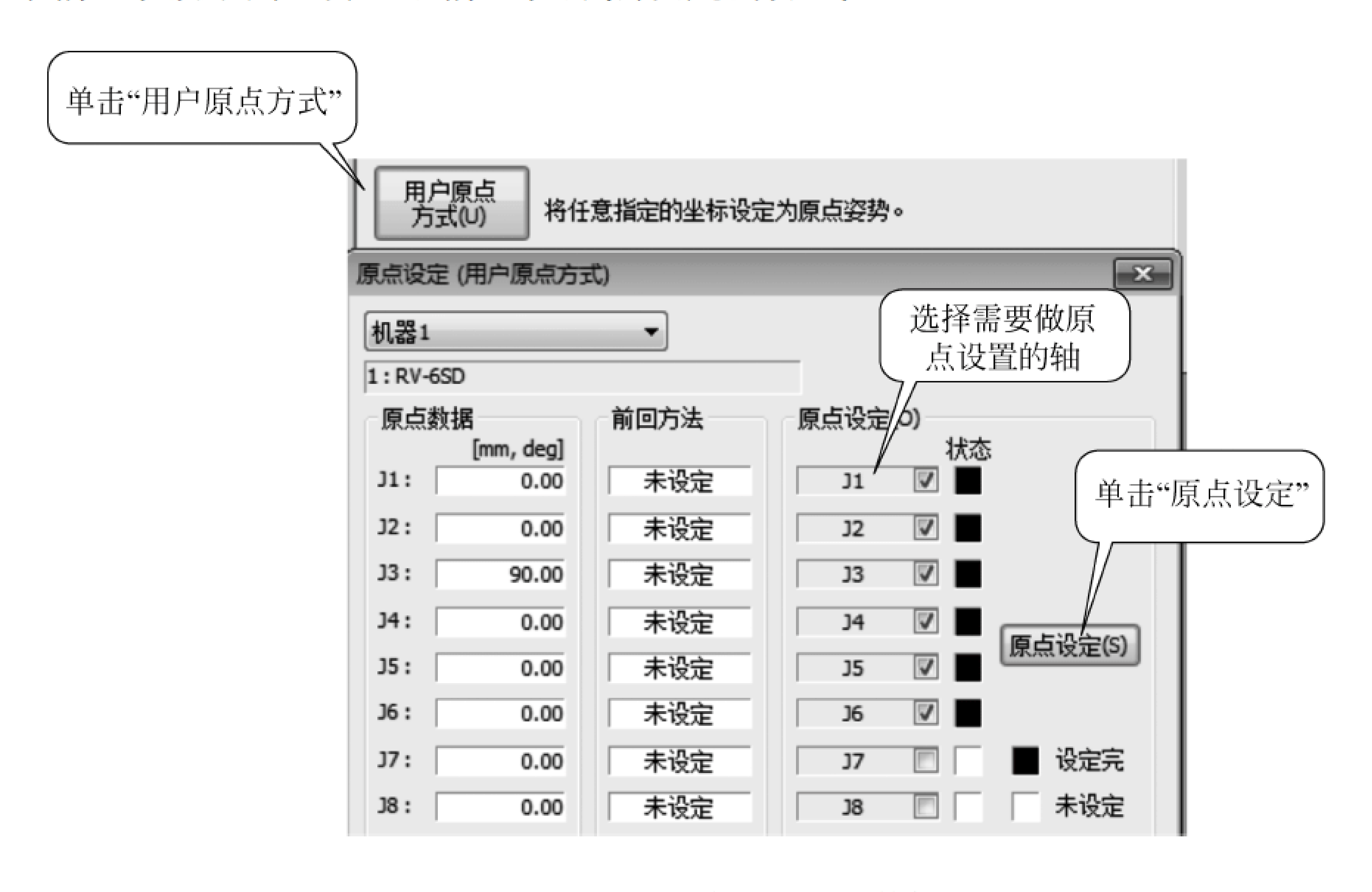


图 30-46　用户原点方式设置原点数据画面

7. 原点参数备份方式

功能：将原点参数备份到计算机。也可以将计算机中的“原点数据”写入到“控制器”，如图 30-47 所示。



图 30-47 “原点参数备份”方式设置原点数据画面

30.5.2 初始化

(1) 功能：将机器人控制器中的数据进行初始化。

可对下列信息进行初始化。

① 时间设定；

② 所有程序的初始化；

③ 电池剩余时间的初始化；

④ 控制器的序列号的确认设定。

(2) 操作方法如图 30-48 所示。



图 30-48 初始化操作框

30.5.3 维修信息预报

功能：将机器人控制器中的维保信息数据进行提示预告。

可对下列维保信息进行提示预告。

(1) 电池使用剩余时间提示预告；

(2) 润滑油使用剩余时间提示预告；

(3) 皮带使用剩余时间的提示预告；

(4) 控制器的序列号的确认设定。

维保信息框如图 30-49 所示。

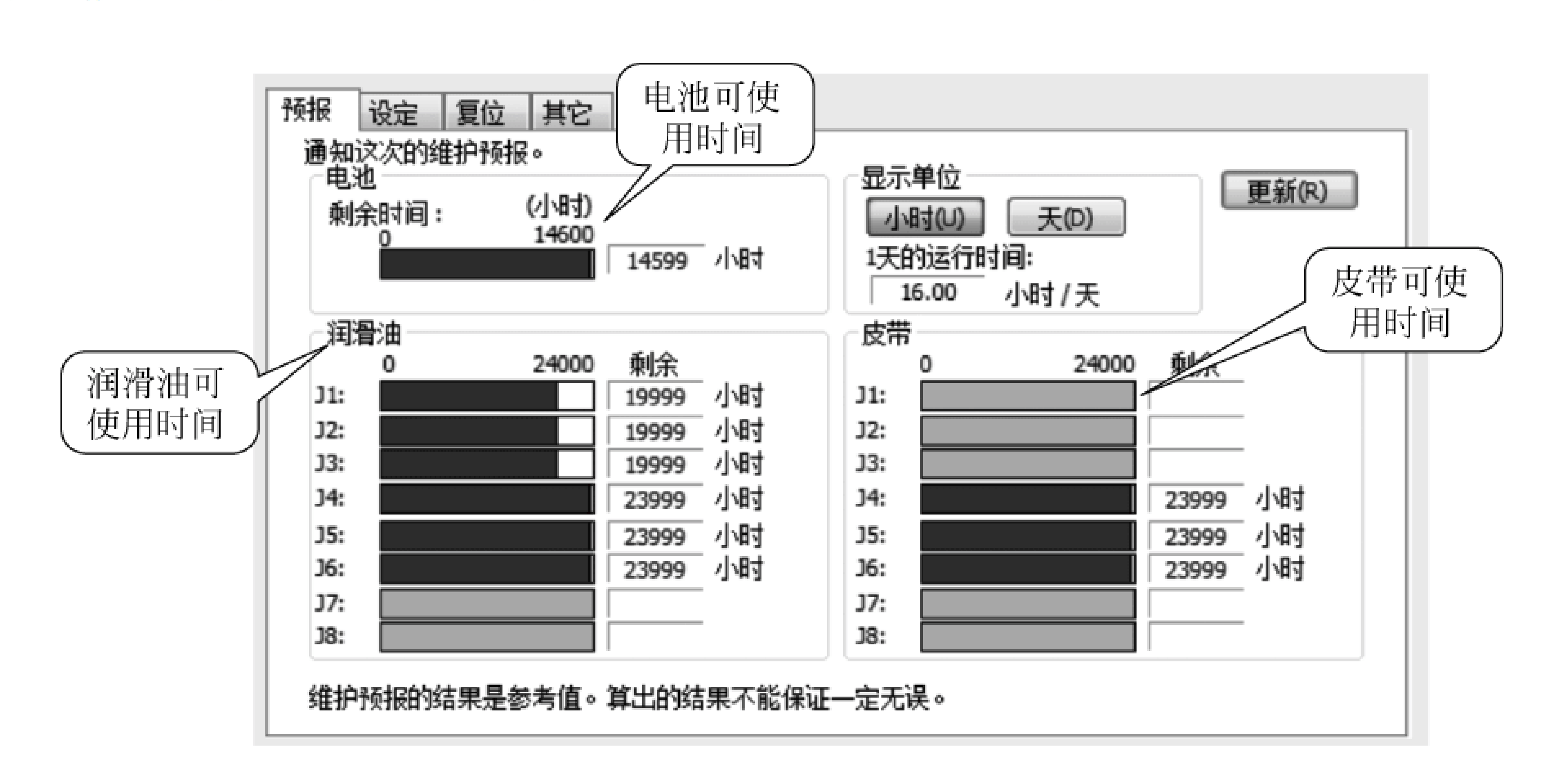


图 30-49 维保信息框

30.5.4 位置恢复支持功能

如果由于碰撞导致抓手变形或由于更换电机导致原点位置发生偏差，使用“位置恢复功能”，只对机器人程序中的一部分位置数据进行“再示教”作业，就可生成补偿位置偏差的参数，对控制器内全部位置数据进行补偿。

30.5.5 TOOL 长度自动计算

功能：自动测定“抓手长度”的功能。在实际安装了“抓手”后，对一个标准点进行 3～8 次的测定，从而获得实际抓手长度，设置为 TOOL 参数(MEXTL)。

30.5.6 伺服监视

功能：对伺服系统的工作状态如电机电流等进行监视。

操作：单击“维护”→“伺服监视”如图 30-50 所示，可以对机器人各轴伺服电机的“位置”“速度”“电流”“负载率”进行监视。图 30-50 中的画面是对电流进行监视。这样可以判断机器人抓取的重量和速度，加减速时间是否达到规范要求。如果电流过大，就要减少抓取工件重量或延长加减速时间。

30.5.7 密码设定

功能：通过设置密码对机器人控制器内的程序、参数及文件进行保护。



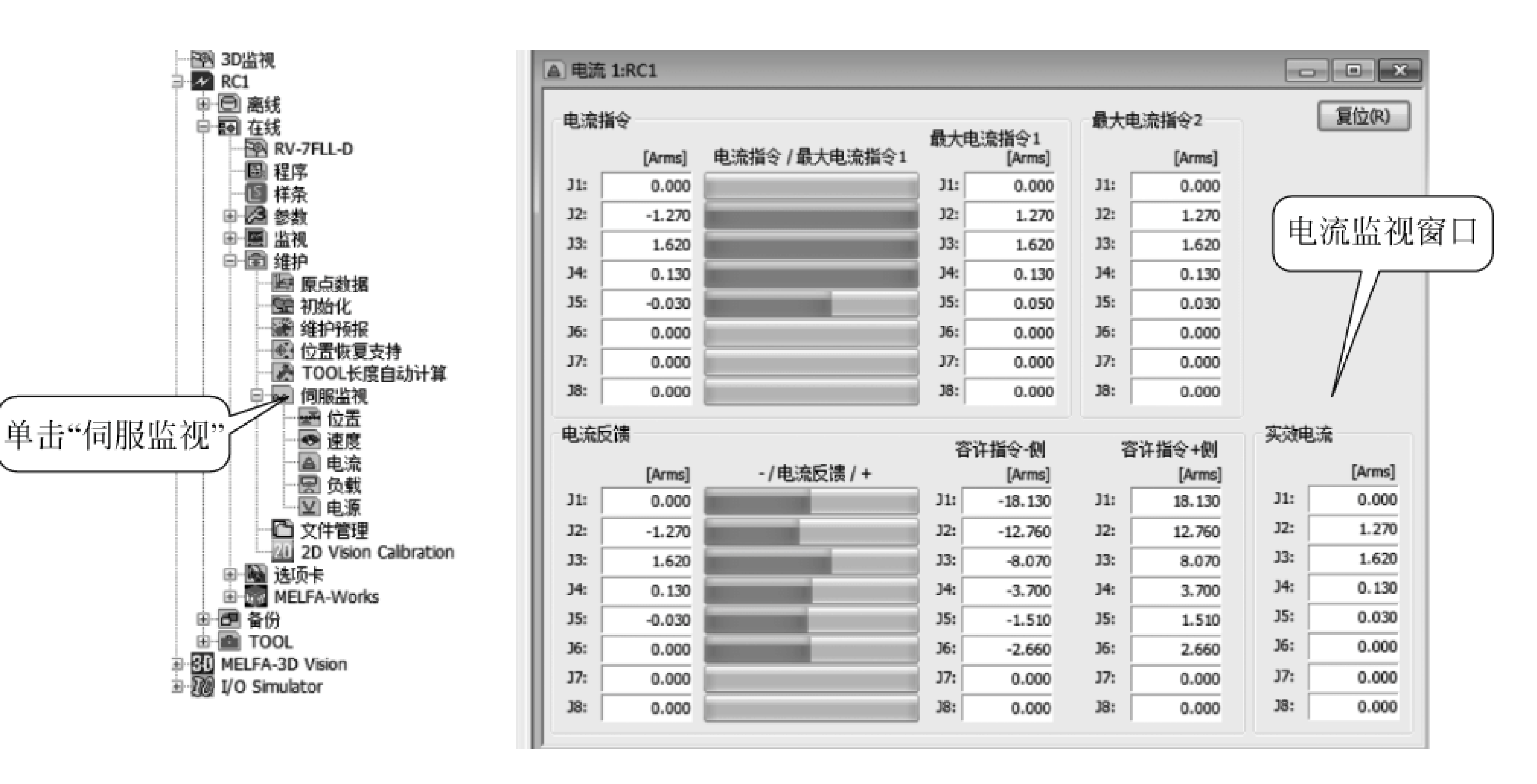


图 30-50　伺服系统工作状态监视画面

30.5.8　文件管理

能够复制、删除、重命名机器人控制器内的文件。

30.5.9　2D 视觉校准

1. 功能

功能：2D 视觉校准功能是标定视觉传感器坐标系与机器人坐标系之间的关系。可以处理 8 个视觉校准数据。

系统构成如图 30-51 所示，执行设备连接。

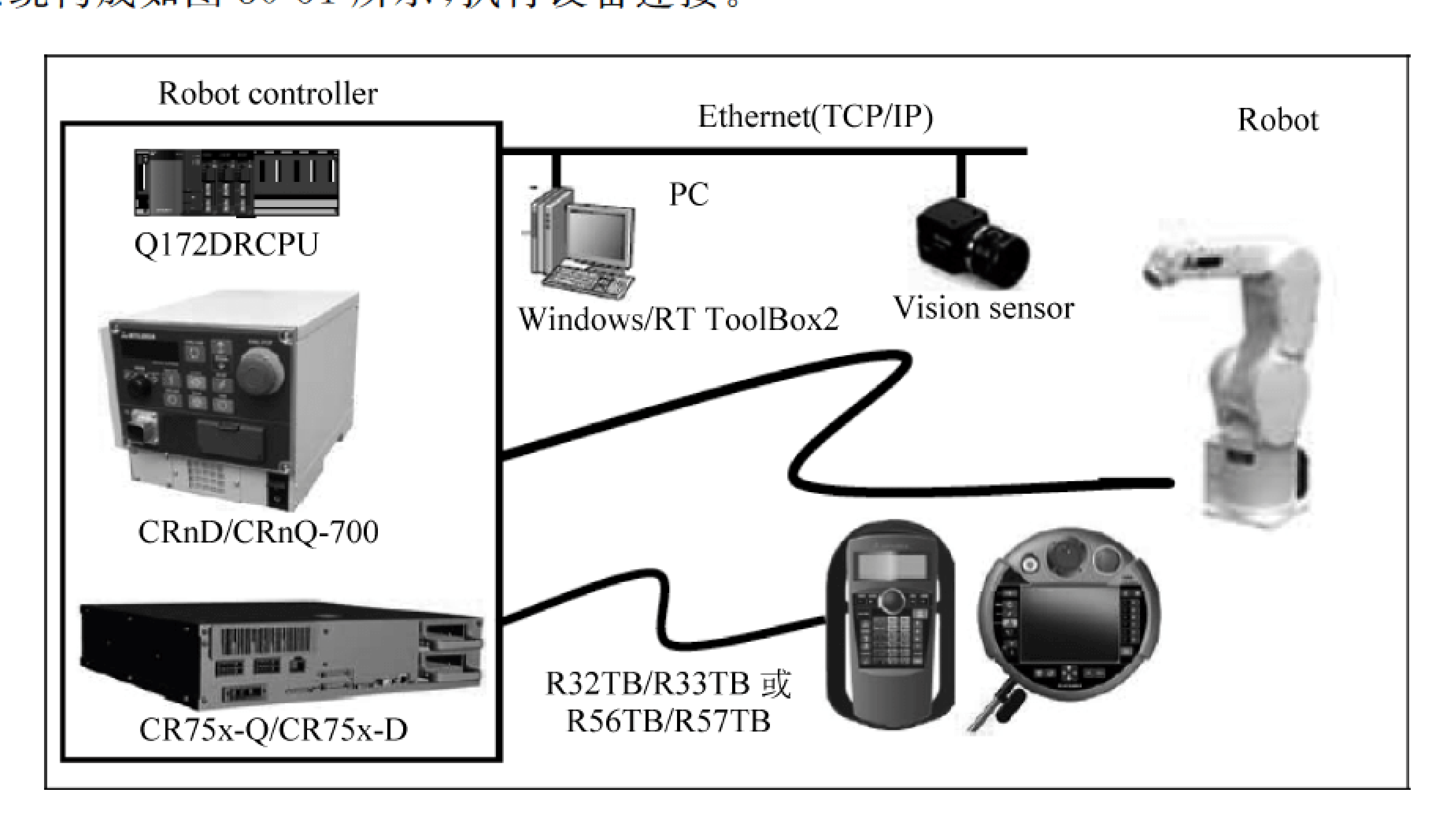


图 30-51　2D 视觉校准时的设备连接

2. 2D 视觉标定的操作程序

(1) 启动 2D 视觉标定，连接机器人。

双击"在线"→"维护"→"2D 视觉标定"。

(2) 选择标定编号。

可选择任一标定编号，最大数为 8，如图 30-52 所示。



图 30-52　选择标定序号

3. 示教点

如图 30-53 所示：

(1) 单击示教点所在行，移动光标，将 TOOL 中心点定位到“标定点”。

(2) 单击 Get the robot position 以获得机器人当前位置。

Robot. X 和 Robot. Y 的数据将自动显示，在 Enabled 框中自动进行检查。

(3) 在单击 Get the robot position 之前，不能编辑示教点数据。

(4) 通过视觉传感器测量“标定指示器”的位置。

分别在(照相机 X)Camera. X 和(照相机 Y)Camera. Y 位置输入 X，Y 像素坐标。

Teaching Points:

	Enabled	No.	Robot.X	Robot.Y	Camera.X	Camera.Y
▸	☑	1	703.680	210.820	100.000	0.000
	☐	2	0.000	0.000	0.000	0.000
	☐	3	0.000	0.000	0.000	0.000
	☐	4	0.000	0.000	0.000	0.000
	☐	5	0.000	0.000	0.000	0.000
	☐	6	0.000	0.000	0.000	0.000

Get the robot position　1　/ 20

图 30-53　获得示教点视觉数据

如果视觉传感器坐标系与机器人坐标系的整合是错误的或示教点过于靠近，则可能出现错误的标定数据。

视觉标定最少需要 4 个示教点，如果是精确标定则需要 9 个点或更多点。分布如图 30-54 所示。

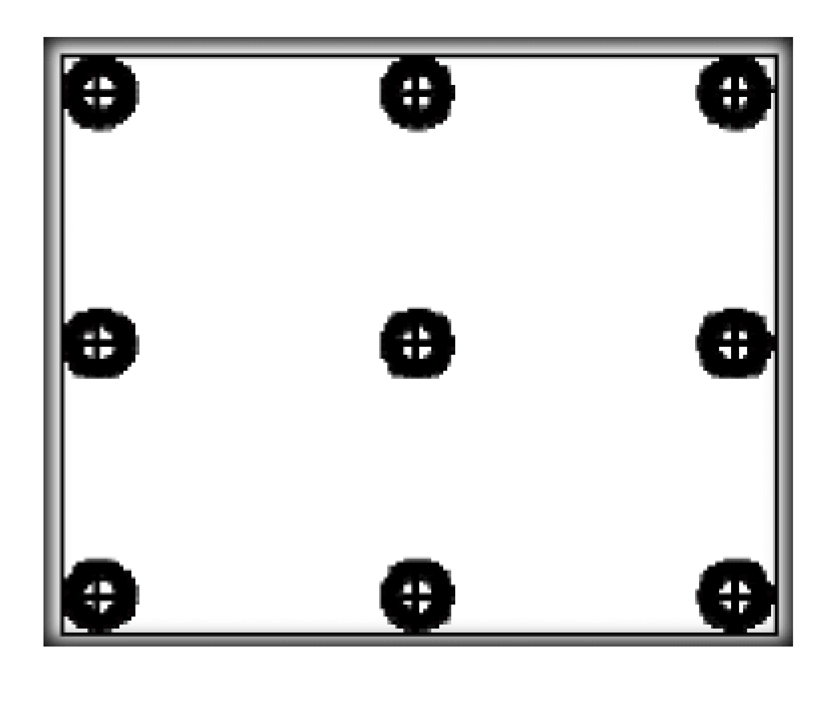


图 30-54　示教点的分布

4. 计算视觉标定数据

当 Teaching points 数据表中已经有 4 个点以上时，Calculate after selecting 4 points or more 按钮就变得有效，单击该按钮，计算结果数据出现在 Result homography matrix 框内，如图 30-55 所示。

5. 写入机器人

单击 Write to robot 按钮，将计算获得的视觉传感器标定数据 VSCALBn 写入控制器。

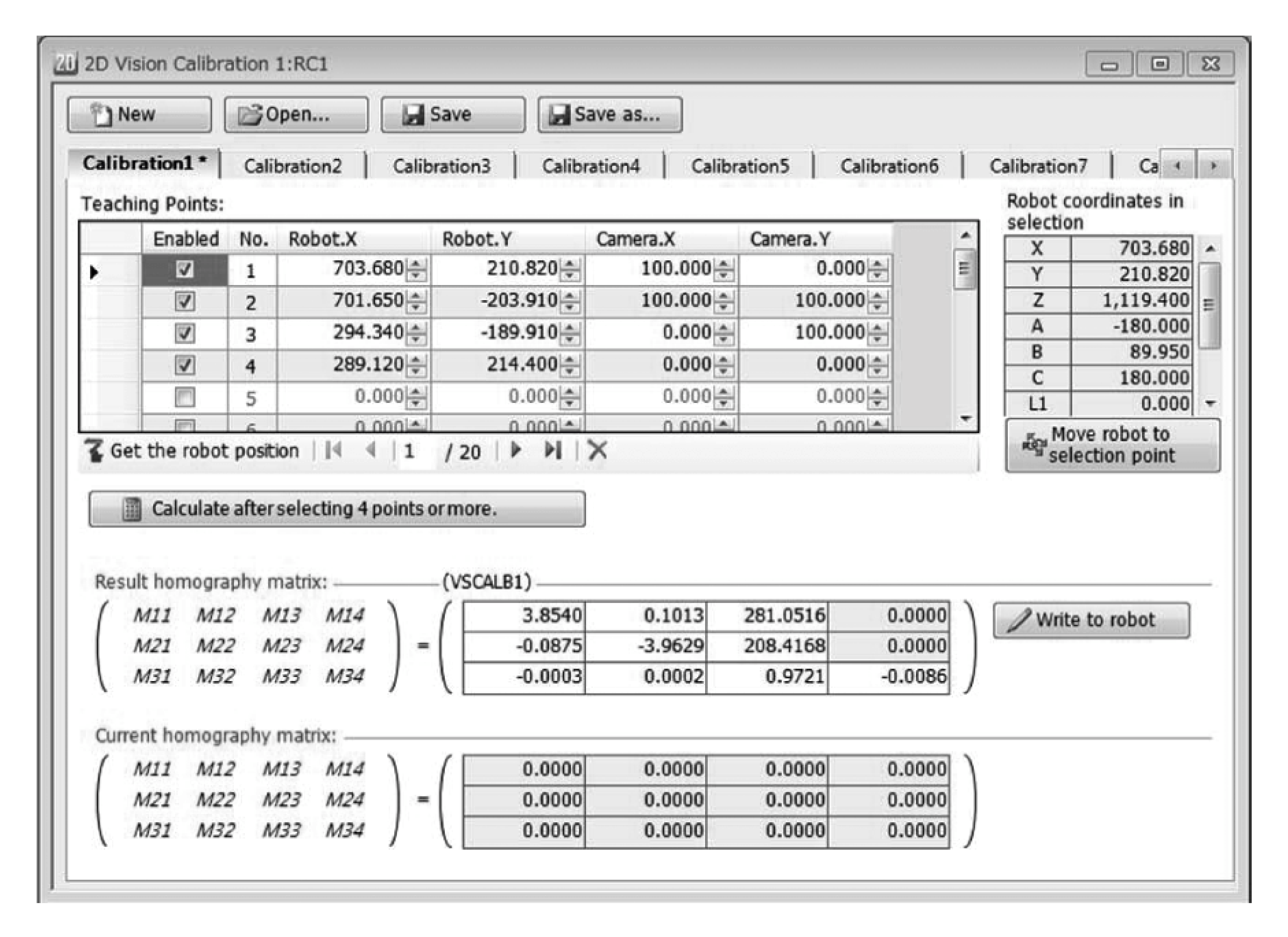


图 30-55　视觉标定计算结果

6. 保存数据

单击 Save 或 Save as 按钮保存示教点和计算结果数据。

30.6　备　　份

1. 功能

将机器人控制器内的全部信息备份到计算机。

2. 操作

单击“在线”→“维护”→“备份”→“全部”，进入全部数据备份画面，如图 30-56 所示。选择“全部”→OK，就将全部信息备份到计算机。



图 30-56　备份操作

30.7 模拟运行

30.7.1 选择模拟工作模式

模拟运行能够完全模拟机器人的所有操作，能够在屏幕上动态地显示机器人运行程序，能够执行 JOG 运行、自动运行、直接指令运行以及调试运行，其功能很强大。

1. 模拟运行的显示

单击“在线”→“模拟”，会弹出以下两个画面，如图 30-57 和图 30-58 所示。

(1) 模拟操作面板；

(2) 3D 运行显示屏。

由于模拟运行状态完全模拟了实际的在线运行状态，所以大部分操作都与“在线状态”相同。

2. 模拟操作面板的操作功能

(1) 选择 JOG 运行模式。

(2) 选择自动运行模式。



图 30-57 模拟操作面板

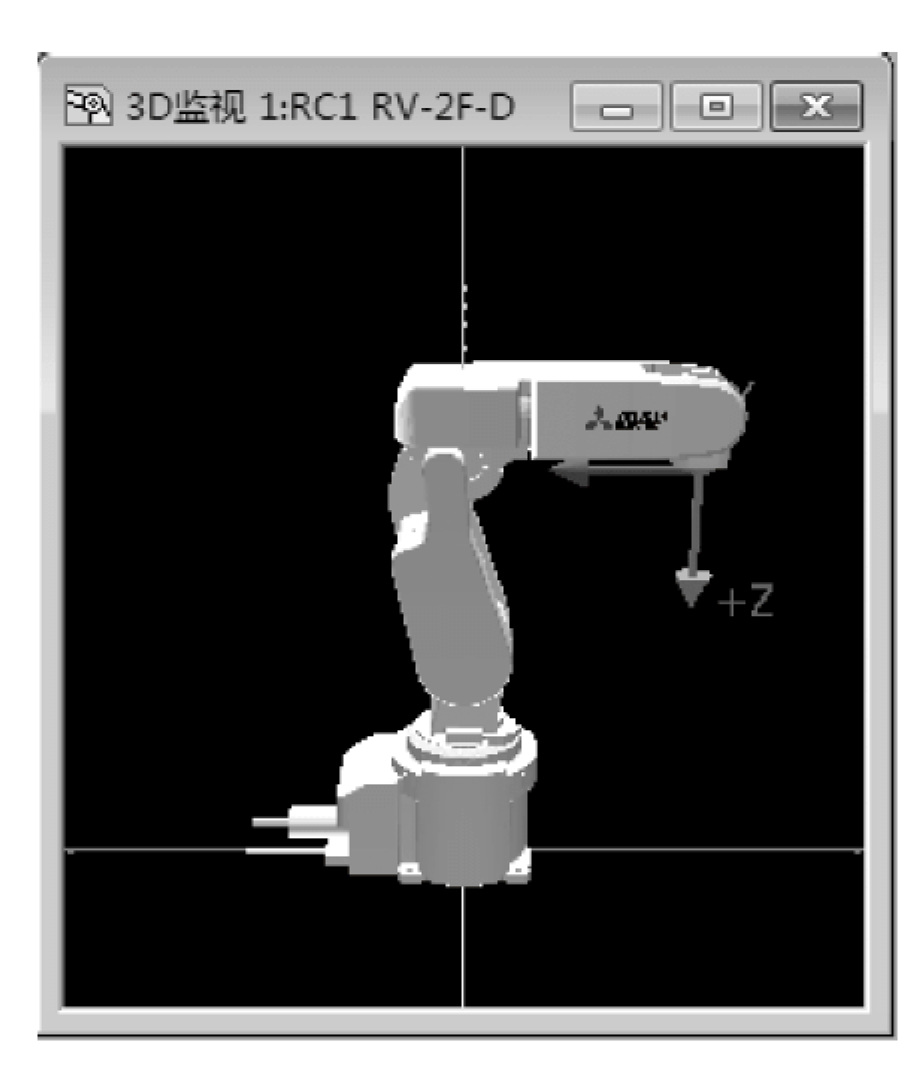


图 30-58 3D 运行显示屏

(3) 选择调试运行模式。

(4) 选择直接运行。

3. 模拟操作面板的监视功能

(1) 显示程序状态并选择程序。

(2) 显示并选择速度倍率。

(3) 显示运行程序。

4. 在工具栏上的图标

在工具栏上的图标其含义如图 30-59 所示。

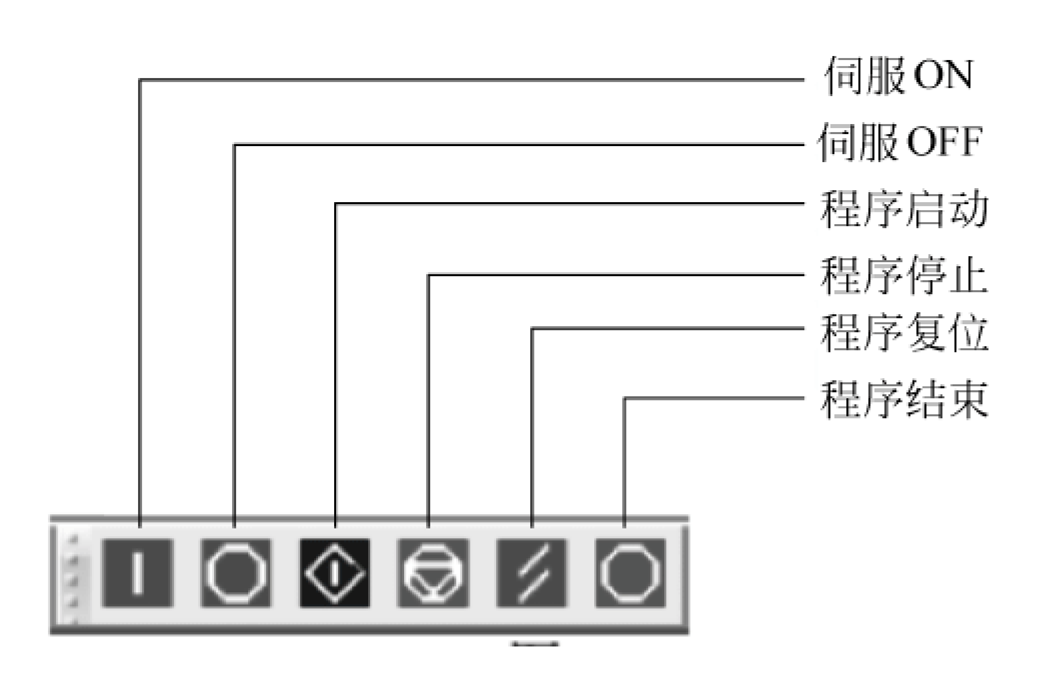


图 30-59　在工具条上的图标

5. 机器人视点的移动

机器人视图(3D监视)的视点,可以通过鼠标操作来变更。具体操作如表30-6所示。

表 30-6　机器人(3D监视)视点的操作方法

要变更的视点	图形上的鼠标操作
视点的旋转	按住左键的同时,左右移动,Z轴为中心的旋转 上下移动,X轴为中心的旋转 按住左+右键的同时,左右移动,Y轴为中心的旋转
视点的移动	按住右键的同时,上下左右移动
图形的扩大/缩小	按住Shift键+左键的同时上下移动

30.7.2　自动运行

1. 程序的选择

1) 如果机器人控制器内有程序

在模拟操作面板上,可以单击"程序选择"图标如图30-60所示,弹出程序选择框,如图30-61所示,选择程序,单击OK按钮,就可以选中程序。



图 30-60　在操作面板上单击"程序选择"



图 30-61　在程序选择框内选择程序

2）如果机器人控制器内没有程序

如果在程序选择框内没有程序，则需要将“离线状态”的程序写入“在线”，操作与常规状态相同，这样在“程序选择框”内就会出现已经写入的程序，这样就有选择对象了。

2. 程序的启动/停止操作

如图 30-62 所示，在模拟操作面板中，有一“操作面板区”，在“操作面板区”内有“伺服ON”“伺服 OFF”“启动”“复位”“停止”“结束”6 个按钮。“操作面板区”用于执行“自动操作”，单击各按钮执行相应的动作。

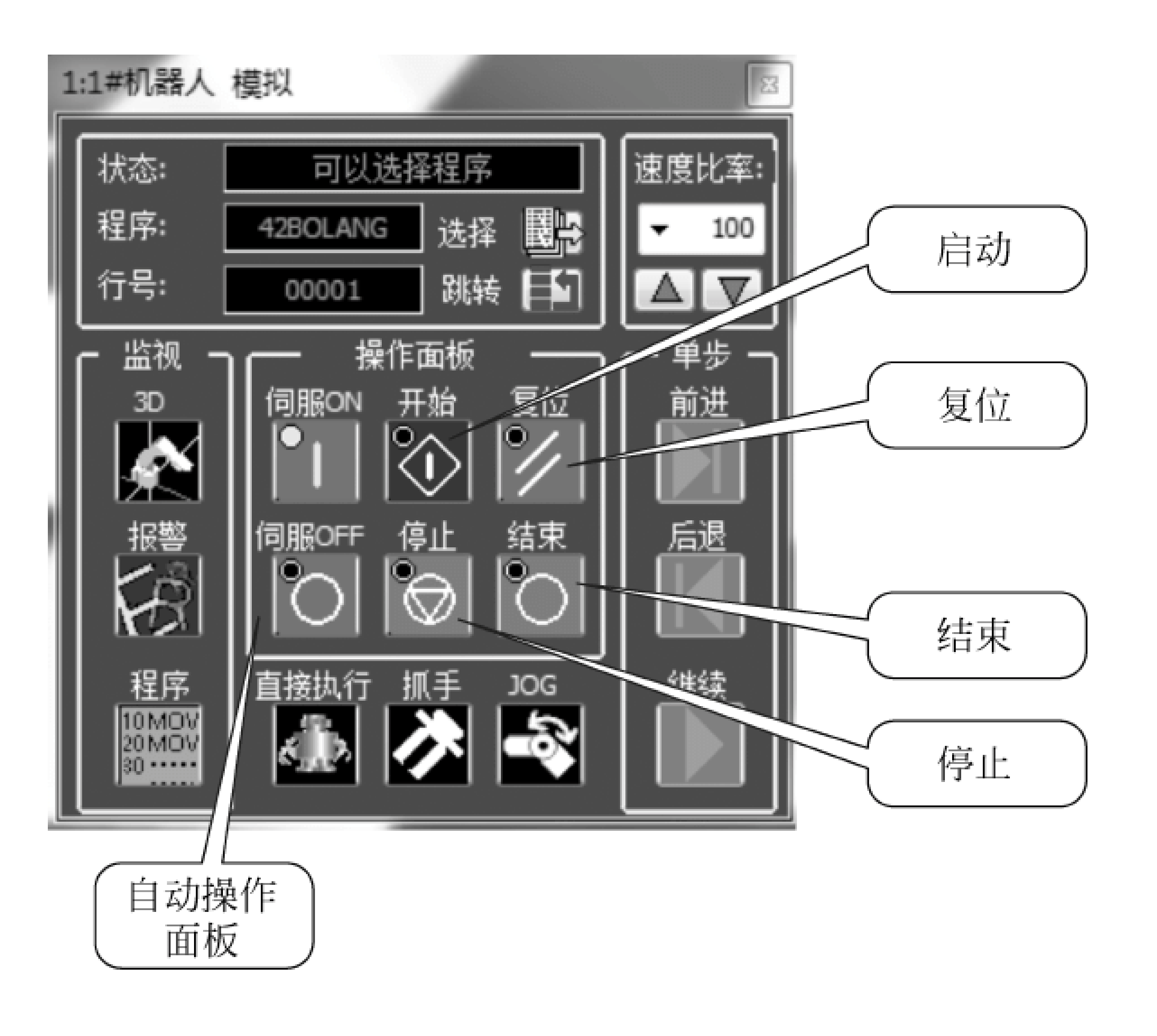


图 30-62　程序的启动停止操作

30.7.3　程序的调试运行

在模拟状态下可以执行调试运行。调试运行的主要形式是“单步运行”。在模拟操作面

板上有单步运行框，如图 30-63 所示。单步运行框内有“前进”“后退”“继续”三个按钮，功能就是单步的前进、后退。与正常调试界面的功能相同。

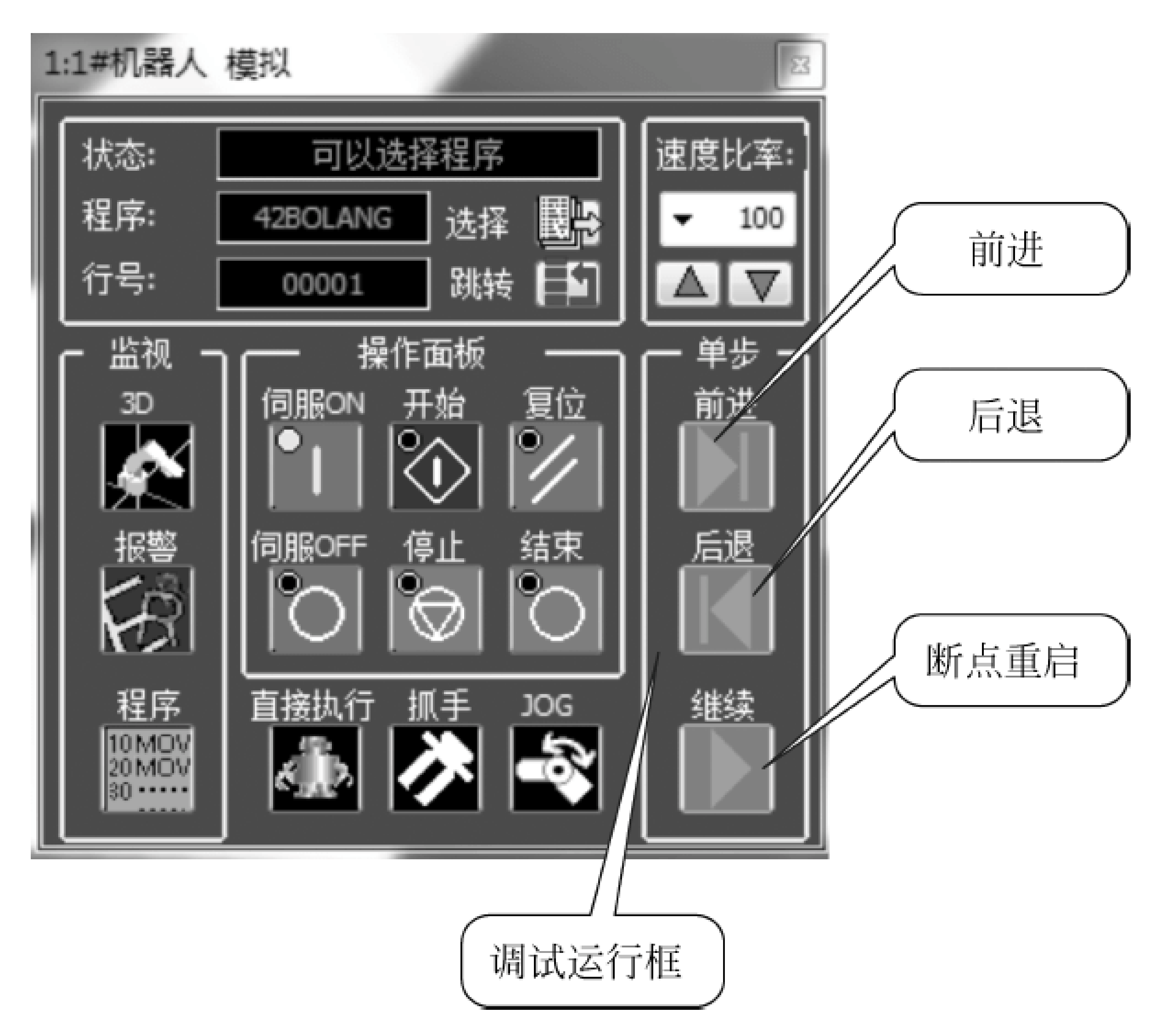


图 30-63 调试功能单步运行操作

30.7.4 运行状态监视

在模拟操作面板上有运行状态监视框，如图 30-64 所示。运行状态监视框内有 3D 显示选择、报警信息选择、当前运行程序界面选择三个按钮。单击不同的按钮将弹出不同的界面。图 30-65 是报警信息界面。

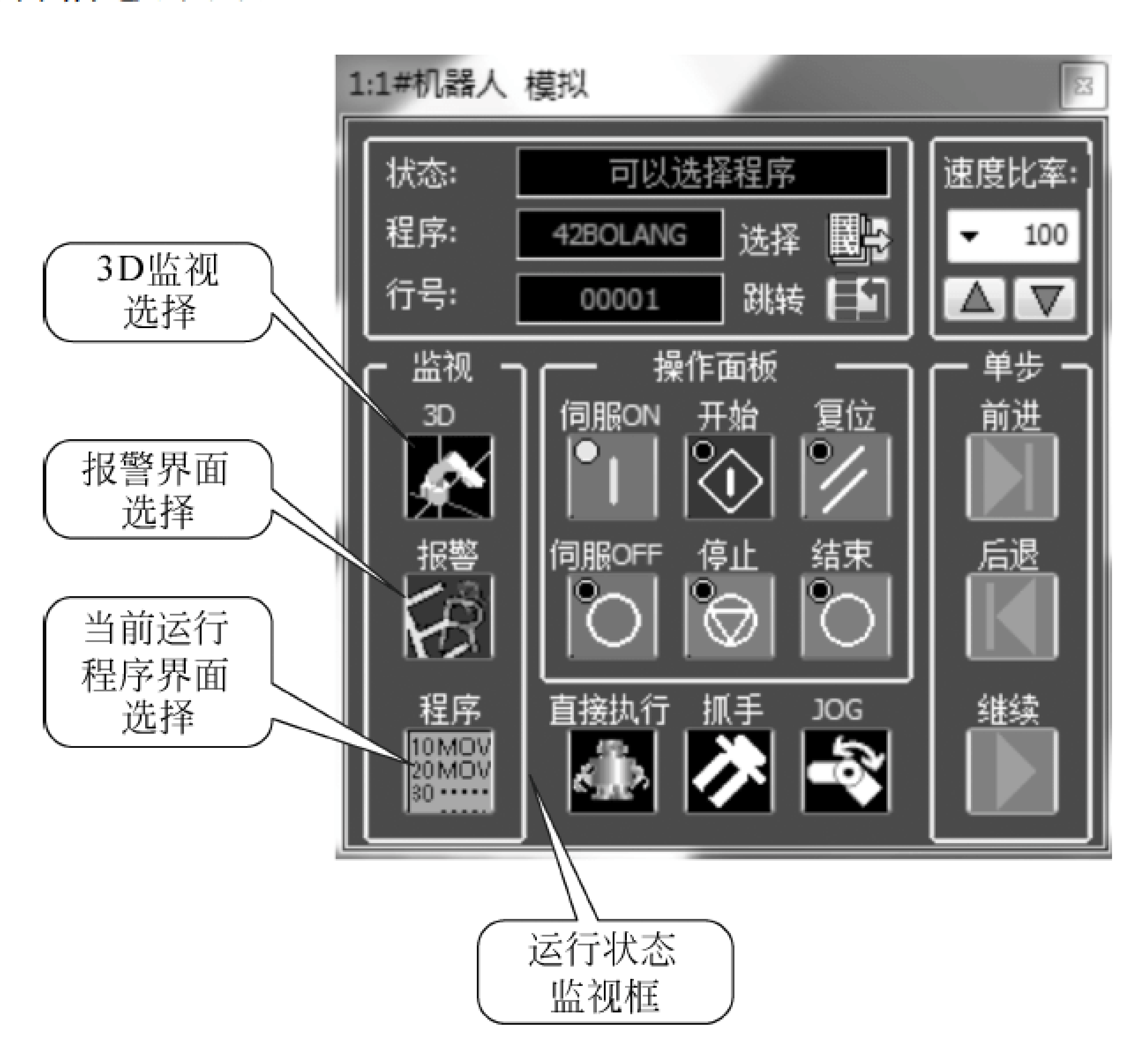


图 30-64 运行状态监视



图 30-65　报警信息窗口

30.7.5　直接指令



图 30-66　选择“直接执行”

直接指令功能是输入或选择某一指令后，直接执行该指令，既不是整个程序的运行，也不是手动操作，而是自动运行的一种形式，在调试时会经常使用。使用“直接运行指令”必须：

(1) 已经选择程序号。

(2) 移动位置点必须是程序中已经定义的“位置点”。

在模拟操作面板上单击“直接执行”图标，如图 30-66 和图 30-67 所示。

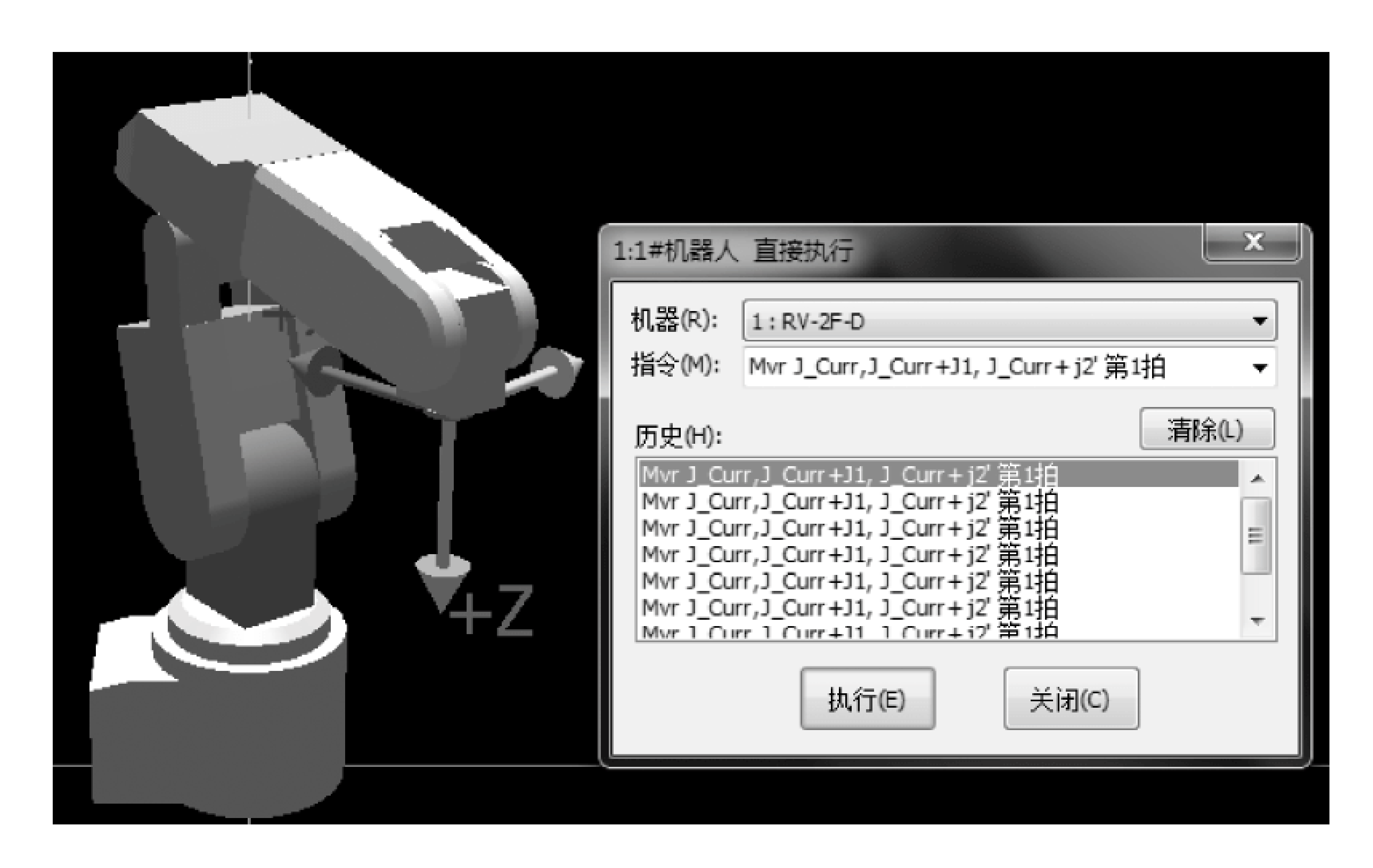


图 30-67　“直接执行”界面

30.7.6　JOG 操作功能

模拟操作面板上有 JOG 操作功能。单击如图 30-68 所示的 JOG 图标就会弹出 JOG 画

面。其操作与示教单元类似。

通过模拟 JOG 操作,可以更清楚地了解各坐标系之间的关系。

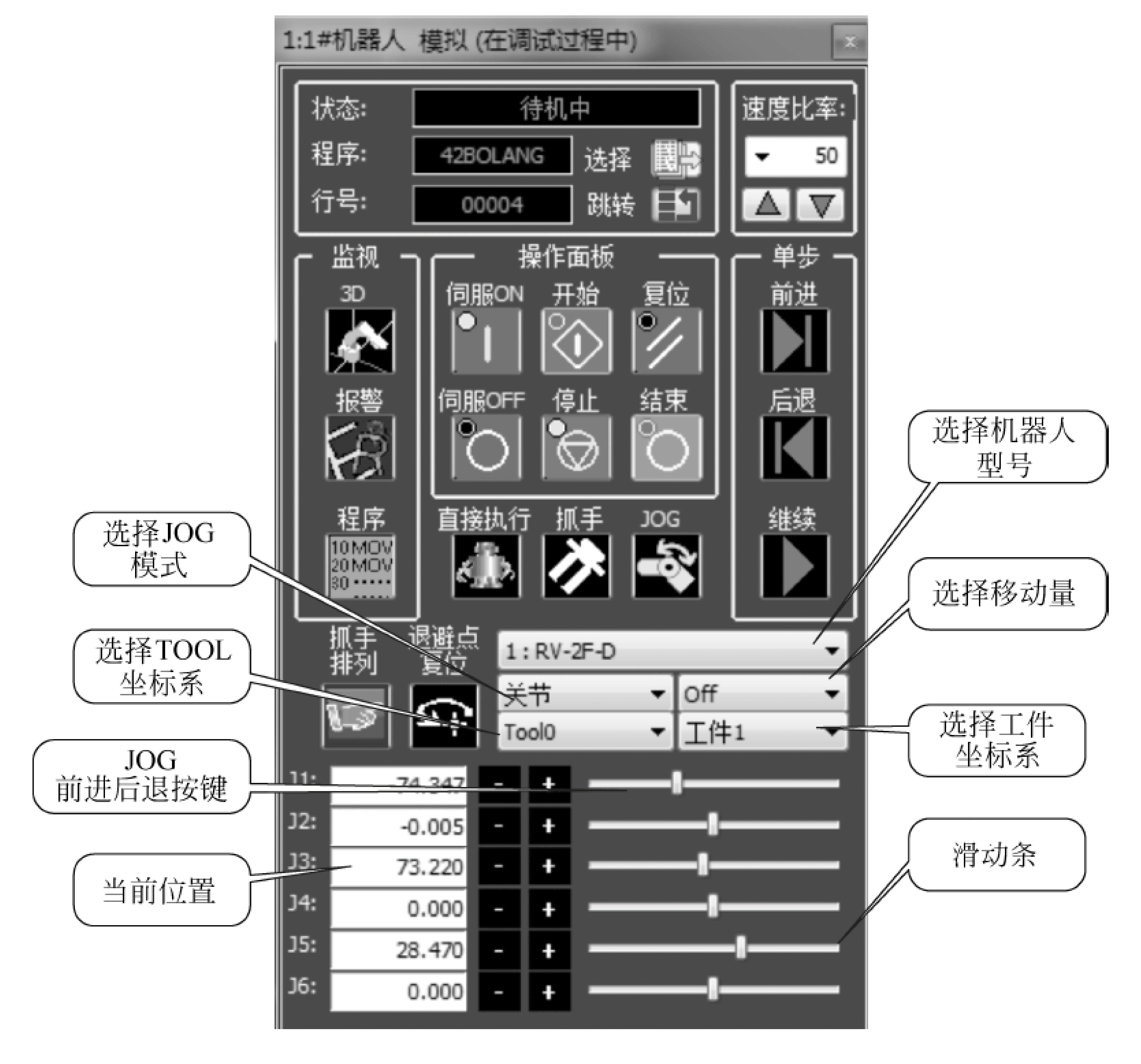


图 30-68　JOG 操作界面

30.8　3D 监视

3D 监视是机器人人性化的一个界面,可以在画面上监视机器人的动作、运动轨迹、各外围设备的相对位置。

在离线状态下,也可以进行 3D 显示。当然最好是在模拟状态下进行 3D 显示。

30.8.1　机器人显示选项

单击菜单上的“3D 显示”→“机器人显示选项”,弹出“机器人显示选项”对话框,如图 30-69 所示,本对话框的功能是选择显示什么内容。

1. 选择“窗口”功能

弹出以下选项。

(1) 是否显示操作面板;

(2) 是否显示工作台面;

(3) 是否显示坐标轴线;

(4) 设置屏幕的背景色。

图 30-69 “机器人显示选项”对话框

2. 选择“机器人模型”

弹出以下选项，根据需要选择：

(1) 是否显示“机器人本体”；

(2) 是否显示“机器人法兰轴(TOOL 坐标系坐标轴)”；

(3) 是否显示抓手；

(4) 是否显示运行轨迹。

3. 样条曲线

显示样条曲线的形状。

30.8.2 布局

“布局”也就是“布置图”。“布局”的功能模拟出外围设备及工件的大小、位置，同时模拟出机器人与外围设备的相对位置。

在本节中，有“零件”及“零件组”的概念。既要对每一零件的属性进行编辑，也要根据需要把相关零件归于“同一组”以方便更进一步制作“布置图”。

系统自带矩形、球形、圆柱等 3D 部件，也可插入其他文件中的 3D 模型图。

单击“3D 显示”→“布局”，弹出“布局一览”对话框，如图 30-70 所示。

“布局一览”对话框中必须设置以下内容。

(1) 零件组：指一组零件，由多个零件组成。可以统一对零件组进行如移动、旋转等编辑。

(2) 零件：某具体的工件。零件可以编辑，如选择为矩形或球形，设置零件大小及在坐标系中的位置。

在“布局一览”对话框，选中要编辑的“零件”，单击“编辑”按钮，弹出如图 30-71 所示“布局编辑”对话框，可进行“零件”名称、组别、位置、大小的编辑。图 30-71 中编辑了一个球形零件，指定了球的大小及位置。

在编辑时，可以在 3D 视图中观察到“零件”的位置和大小。

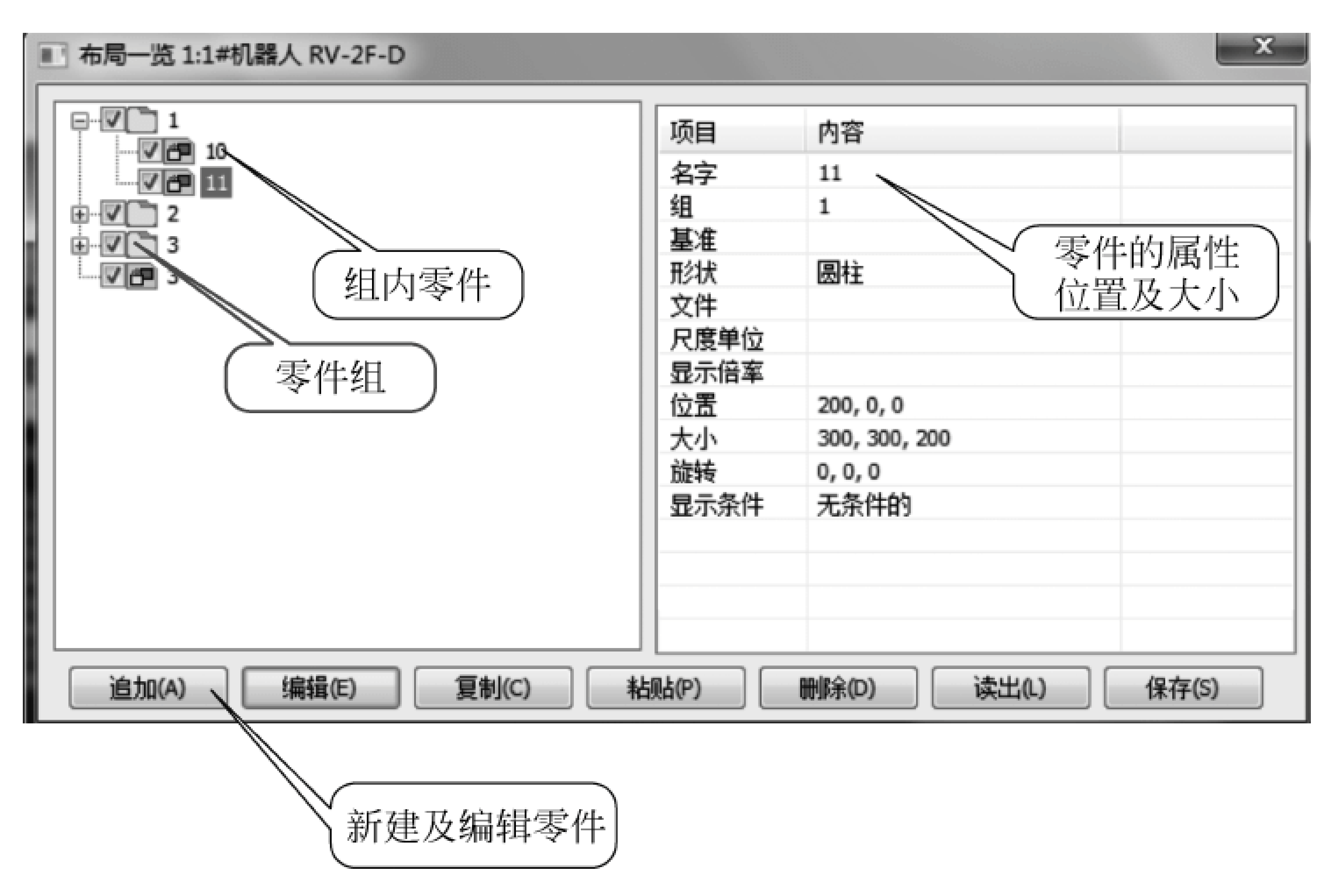


图 30-70 “布局一览”对话框

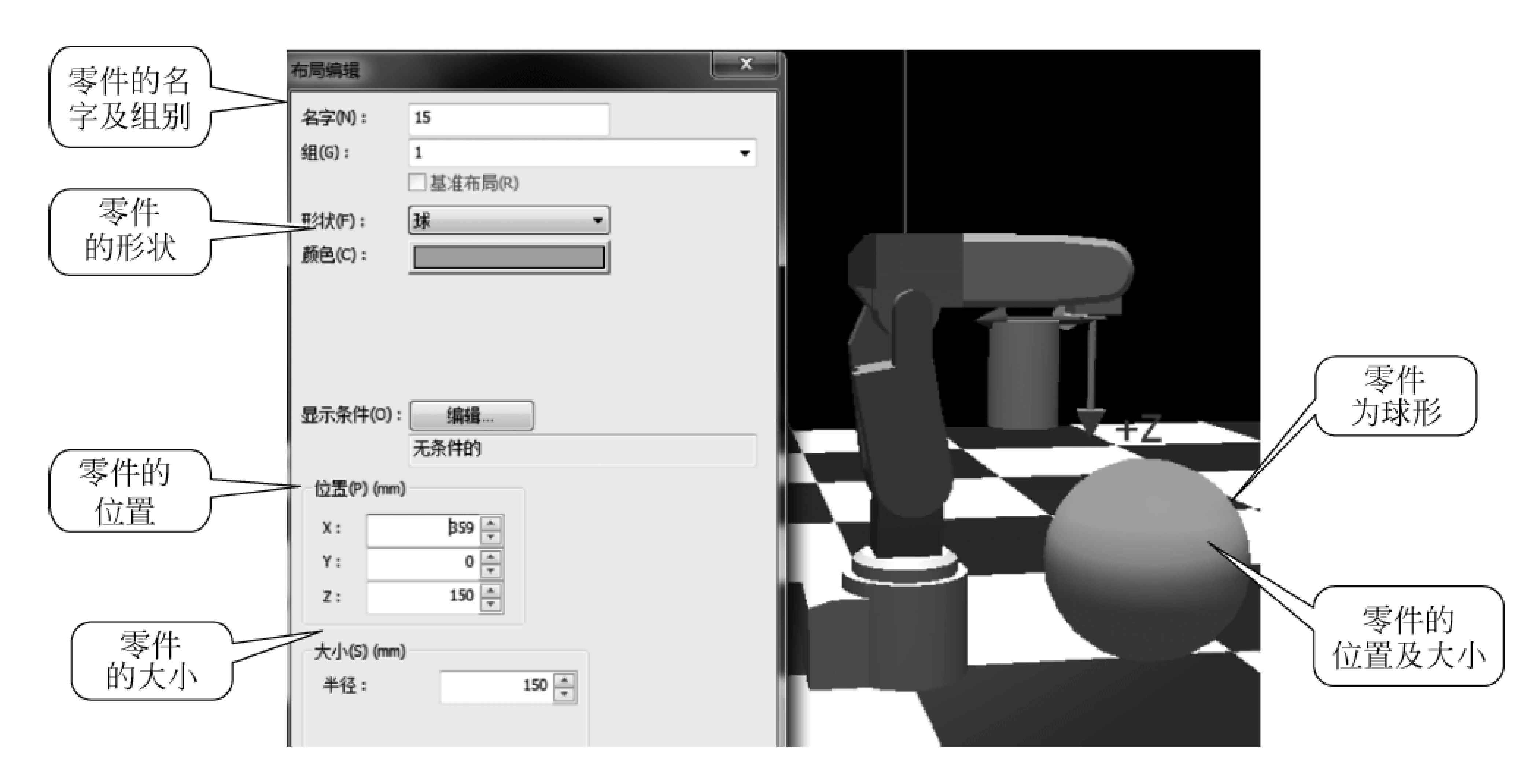


图 30-71 零件的编辑与显示

30.8.3 抓手的设计

1. 抓手设计的功能

抓手是机器人上的附件。本软件提供的抓手设计功能是一个示意功能。抓手的设计与零件的设计相同。先设计抓手的形状大小,在抓手设计画面中的原点位置就是机器人法兰中心的位置。

软件会自动将设计完成的抓手连接在机器人法兰中心。

操作方法如下。

(1) 单击“3D 显示”→“抓手”进入抓手设计画面;

(2) 单击“追加”→“新建”进入一个新抓手文件定义画面;

(3) 单击“编辑”进入抓手的设计画面;

一个抓手可能由多个零部件构成,所以一个抓手也就可以视为一个“零件组”。这样抓手的设计就与零件组的设计相同了。图 30-71 是设计范例。

2. 设计抓手的第一个部件

如图 30-72 所示：

(1) 设置部件名称及组别；

(2) 设计部件的形状和颜色；

(3) 设置部件在坐标系中的位置(坐标系原点就是法兰中心点)；

(4) 设计部件的大小。

设计完成的部件大小及位置如图 30-72 右边所示。

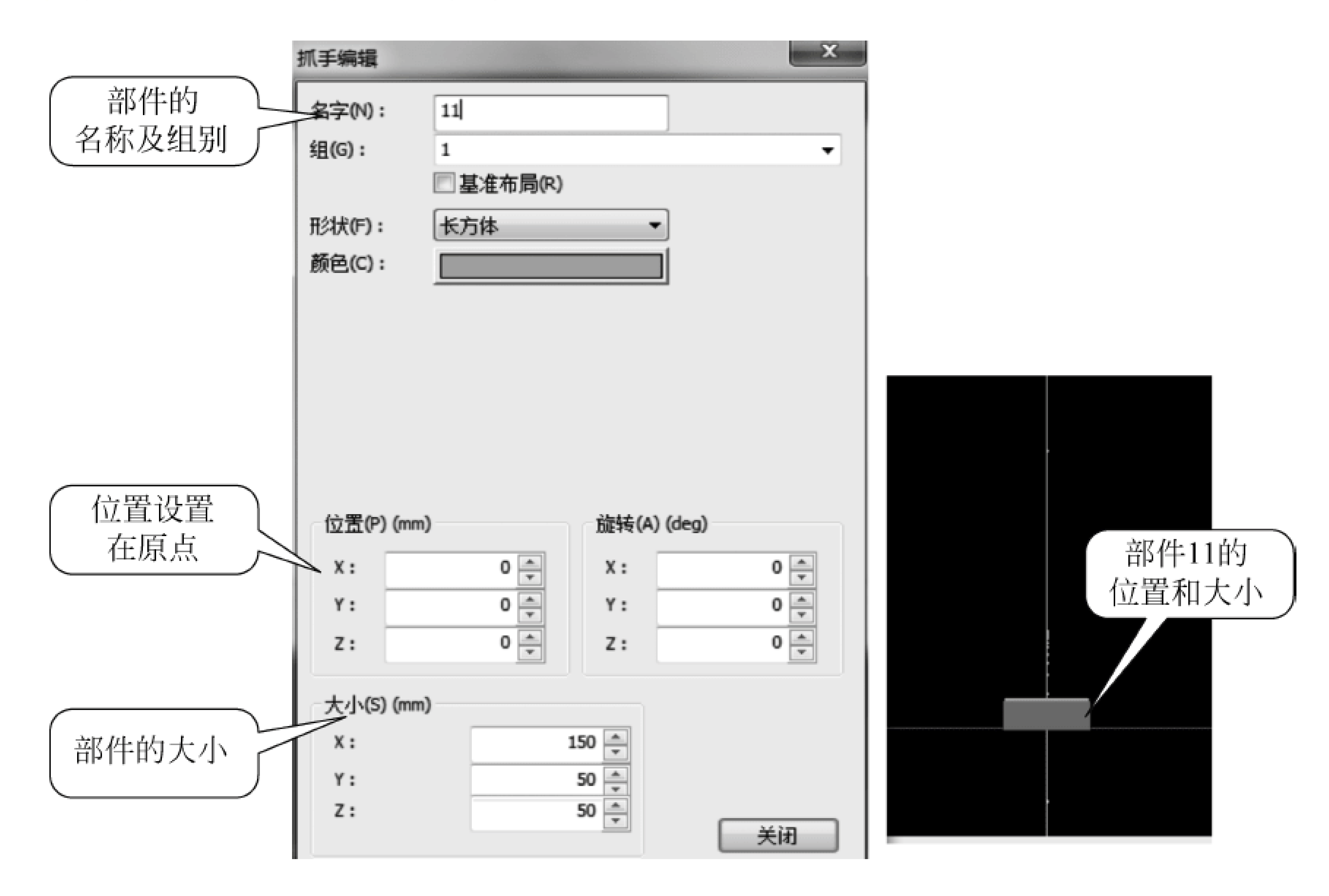


图 30-72 抓手部件 11 的设计

3. 设计抓手的第二个部件

如图 30-73 所示：

(1) 设置部件名称及组别；

(2) 设计部件的形状和颜色；

(3) 部件在坐标系中的位置(坐标系原点就是法兰中心点)；

(4) 设计部件的大小。

设计完成的部件大小及位置如图 30-73 右边所示：第二个部件叠加在第一个部件上。

4. 设计抓手的第三个部件

如图 30-74 所示：

(1) 设置部件名称及组别；

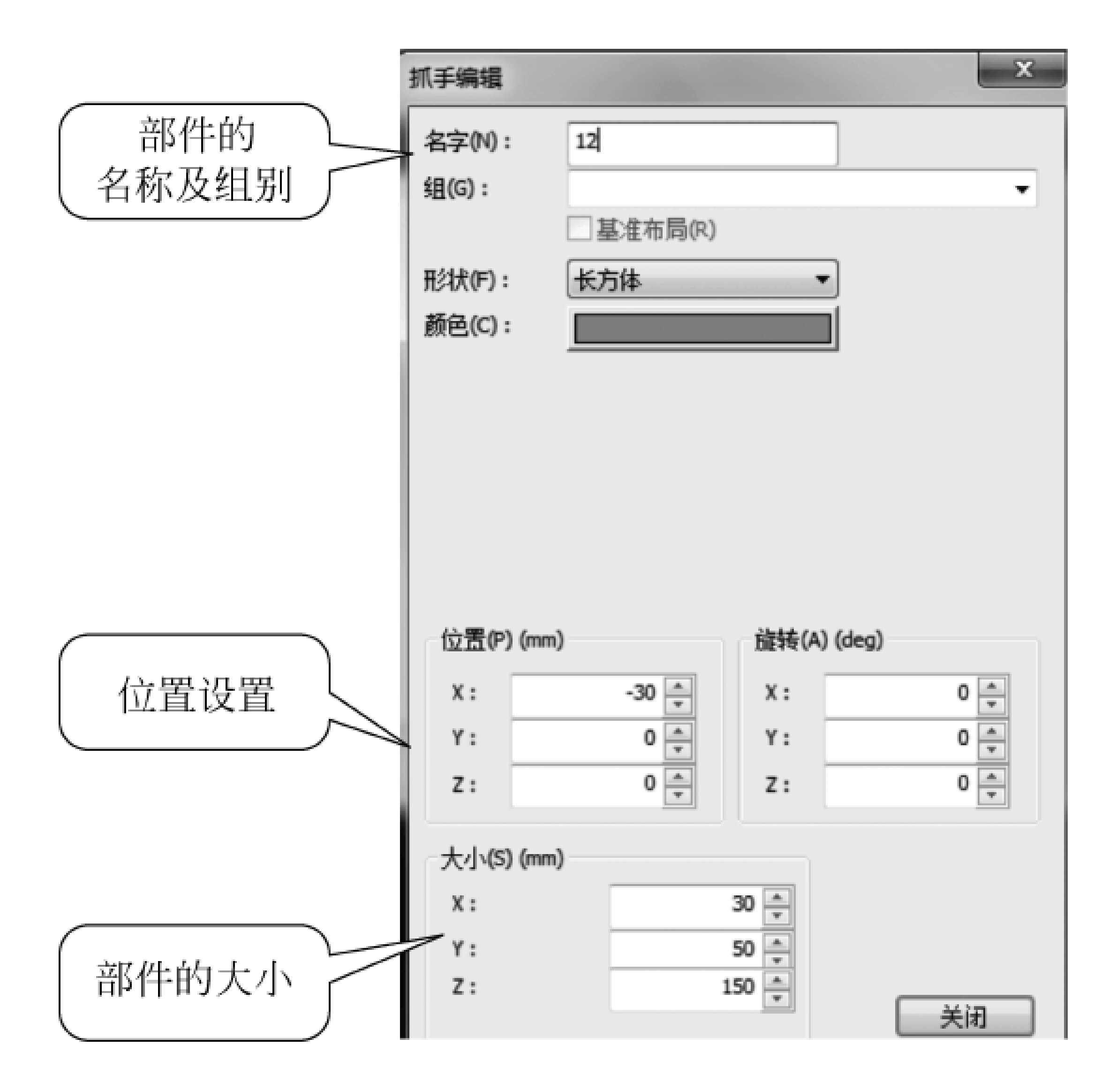
(2) 设计部件的形状和颜色；

(3) 设置部件在坐标系中的位置(坐标系原点就是法兰中心点)；

(4) 设计部件的大小。

设计完成的部件大小及位置如图 30-74 右边所示：第三个部件叠加在第一个部件上。这样就构成了抓手的形状。

将以上文件保存完毕，再回到监视画面，抓手就连接在机器人法兰中心上，如图 30-75 所示。也可以将抓手设计成如图 30-76 所示。



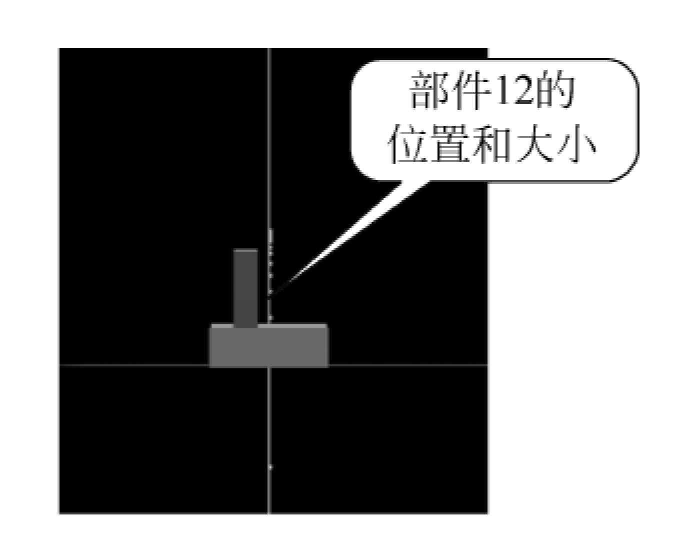
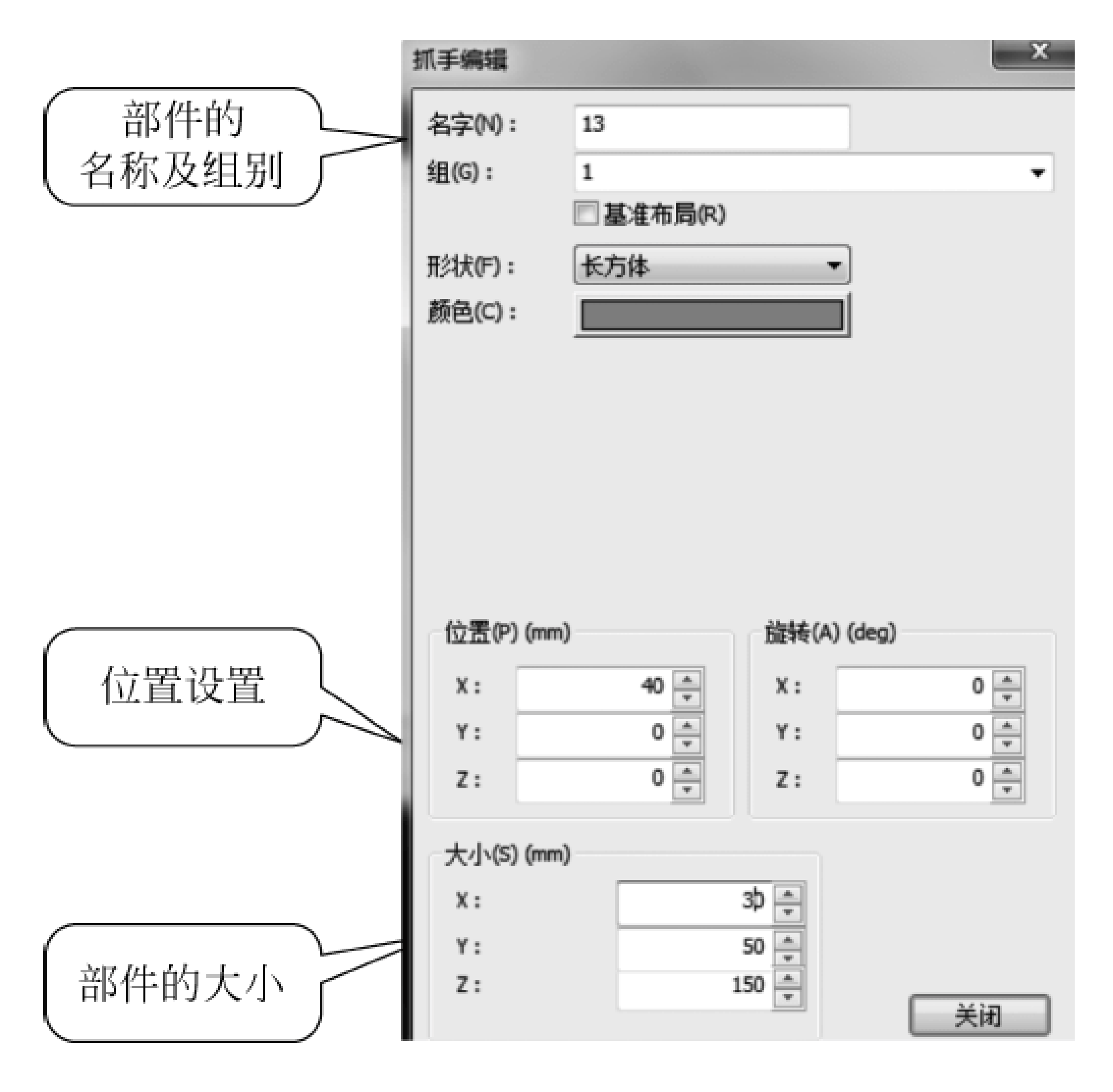


图 30-73　抓手部件 12 的设计



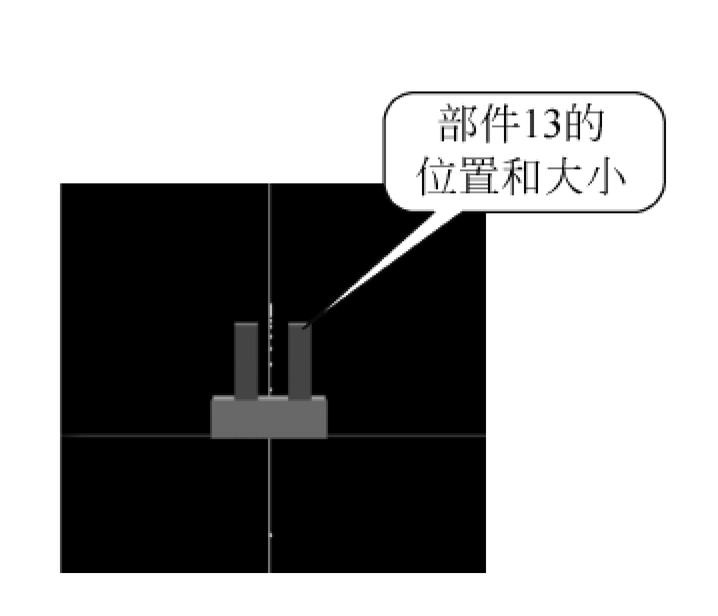


图 30-74　抓手部件 13 的设计

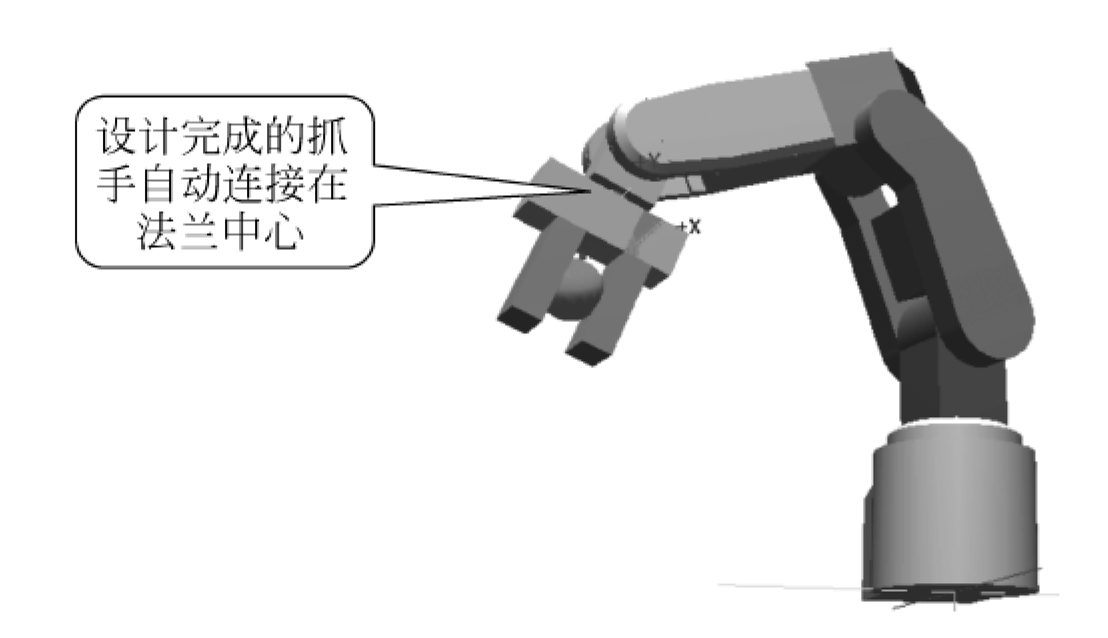


图 30-75　设计安装完成的抓手 1

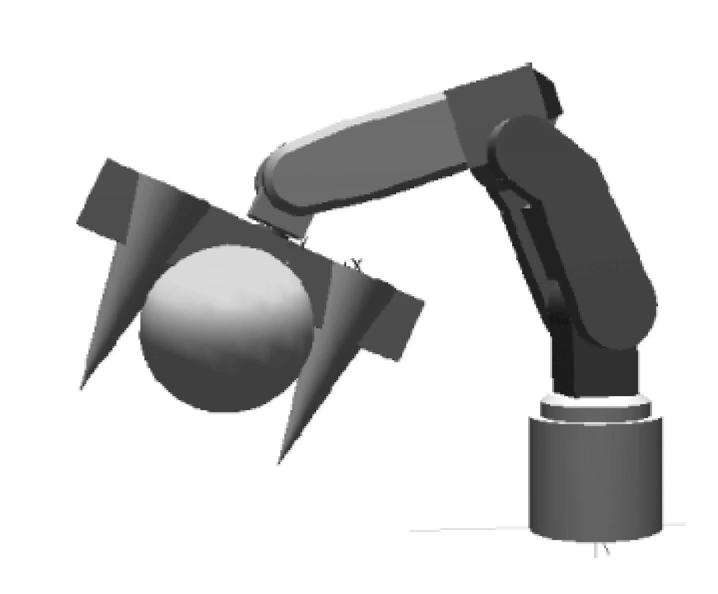


图 30-76　设计安装完成的抓手 2

30.9　需要思考的问题

(1) 如何使用 RT 软件进行参数设置?
(2) 如何使用 RT 软件获取示教的位置点?
(3) 如何使用 RT 软件进行视觉系统的数据标定?
(4) 如何使用 RT 软件进行输入输出信号的监视?
(5) 如何使用 RT 软件进行三维模拟运行?
(6) 如何使用 RT 软件进行程序的调试?

参 考 文 献

1. 黄风.机器人在仪表检测生产线中的应用.金属加工,2016,(18):60-64.
2. 戎罡.三菱电机中大型可编程控制器应用指南.北京:机械工业出版社,2011.
3. 刘伟.六轴工业机器人在自动装配生产线中的应用.电工技术,2015,(8):49-50.
4. 吴昊.基于 PLC 的控制系统在机器人码垛搬运中的应用.山东科学,2011,(6):75-78.
5. 任旭,等.机器人砂带磨削船用螺旋桨关键技术研究.制造技术与机床,2015,(11):127-131.
6. 高强,等.基于力控制的机器人柔性研抛加工系统搭建.制造技术与机床,2015,(10):41-44.
7. 陈君宝.滚边机器人的实际应用.金属加工,2015,(22).60-63.
8. 三菱电机公司.CR750/CR751 控制器操作说明书.2013.
9. 三菱电机公司.GOT Sample Screen Instruction Manualfor SD Series Robot.2011.
10. 三菱电机公司.GOT Direct ConnectionExtended Function Instruction Manual.2011.
11. 三菱电机公司.RT ToolBox2 / RT ToolBox2 mini 操作说明书.2013.
12. 三菱电机公司.RV-4F/7F/13F/20F 系列使用说明书.2013.
13. 三菱电机公司.RV-4F-D/7F-D/13F-D/20F-D 系列标准规格书.2013.
14. 三菱电机公司.三菱电机工业机器人安全手册.2013.
15. 三菱电机公司.CR750/CR751 控制器故障排除操作说明书.2013.
16. 三菱电机公司.Mitsubishi Industrial RobotSD SeriesTracking Function Manual.2009.
17. 三菱电机公司.Mitsubishi Industrial RobotTracking Function Manual CR750/CR751 series controller CRn-700 series controller 2012.
18. 陈先锋.伺服控制技术自学手册.北京:人民邮电出版社,2010.
19. 杨叔子,杨克冲,等.机械工程控制基础.武汉:华中科技大学出版社,2011.
20. 黄风.运动控制器与数控系统的工程应用.北京:机械工业出版社,2014.